Project Synopsis: AI Blog to Podcast Agent with Virtual 3D Avatar

0. Cover

Project title: AI Blog to Podcast Agent with Virtual 3D Avatar

Team name & ID: Team 15

Team Members: Kartik Bansal (Team Lead), Bhaskar Parihar

Institute / Course: GLA UNIVERSITY / B.Tech CSE(AI ML & IOT)

Version: v2.0 (Updated Tech Stack)

Date: February 4, 2026

Revision History

Version	Date	Author	Change
v0.1	Jan 15, 2026	Kartik Bansal	Initial concept and architecture
v1.0	Feb 1, 2026	Team 15	Complete technical specification
v2.0	Feb 4, 2026	Team 15	Updated with optimized FREE tech stack

1. Overview

Problem statement: Content creators and podcasters spend hours manually converting blog posts into audio podcasts, often resulting in monotonous narration that lacks engagement. Existing text-to-speech solutions lack personality and visual engagement, leading to poor listener retention.

Goal: Build an AI-powered autonomous agent that automatically converts blog content into engaging podcast episodes featuring a customizable 3D virtual avatar host with natural voice synthesis, visual animations, and seamless content adaptation - using entirely FREE and open-source technologies.

Non-goals: Live streaming capabilities, multi-avatar conversations, real-time user interaction, or monetization features in v1.

Value proposition: Reduce podcast production time from hours to minutes with AI-driven content transformation, natural voice narration, and engaging 3D avatar presentation - all at ZERO cost using free-tier services with generous limits.

2. Scope and Control

2.1 In-scope

Blog content ingestion from URL, text input, and markdown files, AI-powered content analysis and podcast script generation using Groq API (14,400 req/day FREE), advanced text-to-speech with gTTS (unlimited FREE), customizable Ready Player Me 3D avatars with lip-sync, real-time WebSocket updates, video export in MP4 format, Next.js 14 web dashboard with Three.js 3D preview, and analytics tracking.

2.2 Out-of-scope

Live streaming capabilities, multi-language support beyond English in v1, custom avatar modeling tools, real-time collaboration features, direct publishing to podcast platforms, and paid premium features.

2.3 Assumptions

Blog content is primarily text-based and publicly accessible.

Users have modern browsers supporting WebGL for 3D avatar preview.

Free tier limits are sufficient for development and initial user base.

WebSocket connections remain stable for real-time updates.

2.4 Constraints

12-week development timeline with 2-person team.

ZERO budget constraint - 100% FREE services only.

Groq API: 14,400 requests/day limit.

Hugging Face Spaces: 16GB RAM, CPU-based processing.

Cloudflare R2: 10GB storage limit.

Supabase: 500MB database limit.

Maximum podcast length of 20 minutes per episode in v1.

2.5 Dependencies

Groq API - Free LLM for content analysis and script generation (14,400 req/day).

gTTS (Google Text-to-Speech) - Unlimited free voice synthesis.

Whisper (base model) - Free speech-to-text for any audio input.

Ready Player Me - Free 3D avatar platform with customization.

Supabase - Free PostgreSQL database (500MB).

Upstash Redis - Free caching service (10k req/day).

Cloudflare R2 - Free object storage (10GB).

Vercel - Free frontend hosting (unlimited).

Hugging Face Spaces - Free backend hosting (16GB RAM).

FFmpeg - Open-source video processing.

2.6 Acceptance Criteria and Sign-off

GIVEN a blog URL **WHEN** submitted to the system **THEN** a podcast script is generated within 30 seconds using Groq API with ≥85% content relevance.

GIVEN a generated script **WHEN** gTTS synthesis is applied **THEN** audio is generated within 10 seconds with clear pronunciation.

GIVEN audio and script **WHEN** Ready Player Me avatar rendering begins **THEN** lip-sync accuracy is ≥85% and video renders in <5 minutes for 10-minute content.

GIVEN a user is viewing the dashboard **WHEN** podcast generation is in progress **THEN** real-time progress updates appear via WebSocket within 2 seconds.

Sign-off: Team Lead approves final demo, all P0/P1 issues resolved, mentor approval received, system runs entirely on FREE services.

Sign-off Table

Stakeholder	Role	Decision Area	Approval	Date
Kartik Bansal	Team Lead	Overall project, architecture	Approved	Feb 4
Project Mentor	Mentor	Final acceptance	Pending	TBD

3. Stakeholders and RACI

Activity	Responsible (R)	Accountable (A)	Consulted/Informed
Architecture & Design	Kartik	Kartik	Preshit, Mentor
Backend (FastAPI)	Kartik	Kartik	Preshit
AI Integration (Groq, gTTS)	Kartik	Kartik	Preshit
Frontend (Next.js)	Bhaskar	Kartik	Preshit
3D Avatar (Three.js, RPM)	Bhaskar	Kartik	Preshit
WebSocket Integration	Both	Kartik	Preshit
Testing & Deployment	Both	Kartik	Preshit

4. Team and Roles

Member	Role	Responsibilities	Key Skills	Availability	Contact
Kartik Bansal	Team Lead & Backend Dev	FastAPI backend, Groq/Whisper/gTTS integration, Supabase DB, Redis caching, HF Spaces deployment, WebSocket server	Python, FastAPI, AI APIs, PostgreSQL, WebSockets	25 hrs/wk	8273889824
Bhaskar Parihar	Frontend & 3D Dev	Next.js 14 app, Three.js/R3F integration, Ready Player Me avatars, Socket.io client, Vercel deployment, UI/UX	Next.js, TypeScript, Three.js, React, WebSockets	25 hrs/wk	97194 33175

Role Distribution Rationale: Kartik handles all backend services including FastAPI development, AI service integrations (Groq, Whisper, gTTS), database management, and deployment to Hugging Face Spaces. Bhaskar focuses on the Next.js frontend, 3D avatar visualization using Three.js and React Three Fiber, Ready Player Me integration, and real-time WebSocket communication. Both collaborate on testing and optimization.

5. Week-wise Plan and Assignments

Development schedule for Feb-Apr 2026 (optimized for FREE tech stack):

Week	Dates	Milestones	Kartik (Backend/AI)	Bhaskar (Frontend/3D)	Status
1	3-9 Feb	Setup & Architecture	FastAPI setup, Groq API test	Next.js 14 setup, Tailwind	Planned
2	10-16 Feb	Database & Auth	Supabase setup, Redis config	Auth UI, routing	Planned
3	17-23 Feb	Content Processing	Blog parser, Groq integration	Input forms, validation	Planned
4	24 Feb-2 Mar	Script Generation	Script gen with Groq API	Script editor component	Planned
5	3-9 Mar	Voice Synthesis	gTTS integration, audio gen	Audio player UI	Planned
6	10-16 Mar	3D Avatar Setup	RPM API integration	Three.js setup, R3F basics	Planned
7	17-23 Mar	Avatar Rendering	Lip-sync logic, FFmpeg	RPM avatar loader, controls	Planned
8	24-30 Mar	WebSocket Real-time	Socket.io server setup	Socket.io client, live updates	Planned
9	31 Mar-6 Apr	Video Pipeline	FFmpeg pipeline, R2 upload	3D scene rendering	Planned
10	7-13 Apr	Integration	End-to-end testing	UI polish, animations	Planned
11	14-20 Apr	Optimization	Caching, performance tuning	Load optimization	Planned
12	21-27 Apr	Final Release	HF Spaces deploy, docs	Vercel deploy, demo prep	Planned

9. System Architecture

9.1 Complete Tech Stack (100% FREE)

FRONTEND (Deployed on Vercel - FREE, Unlimited)

Technology	Purpose	Cost
Next.js 14 (App Router)	Main framework with TypeScript	FREE
Tailwind CSS	Utility-first styling	FREE
Three.js + React Three Fiber	3D rendering and avatar viewer	FREE
Socket.io-client	Real-time WebSocket communication	FREE

BACKEND (Deployed on Hugging Face Spaces - FREE, 16GB RAM)

Technology	Purpose	Cost
Python 3.11 + FastAPI	REST API framework	FREE
Uvicorn	ASGI server	FREE
Socket.io (Python)	WebSocket server for real-time	FREE
FFmpeg	Video processing and encoding	FREE (open-source)

AI SERVICES (All FREE with Generous Limits)

Service	Purpose	Free Tier Limit
Groq API	LLM for script generation	14,400 requests/day
Whisper (base model)	Speech-to-text conversion	Unlimited (runs on server)
gTTS	Text-to-speech synthesis	Unlimited

DATABASE & STORAGE (All FREE)

Service	Purpose	Free Tier Limit
Supabase	PostgreSQL database	500MB storage
Upstash Redis	Caching and job queue	10,000 requests/day
Cloudflare R2	Object storage for videos	10GB storage

3D AVATAR PLATFORM

Service	Purpose	Cost
Ready Player Me	Customizable 3D avatars with GLB export	FREE

9.2 High-Level Architecture

Frontend Layer (Vercel):

• Next.js 14 with App Router and TypeScript for type-safe development

• Tailwind CSS for responsive, modern UI design

• Three.js + React Three Fiber for real-time 3D avatar preview

• Socket.io-client for WebSocket connections to backend

Backend Layer (Hugging Face Spaces):

• FastAPI (Python 3.11) for REST API endpoints

• Uvicorn ASGI server for high-performance async handling

• Socket.io server for real-time progress updates

• Background task workers for async processing

AI Processing:

• Groq API for ultra-fast LLM inference (script generation)

• Whisper base model for speech-to-text (runs on HF Spaces)

• gTTS for natural-sounding text-to-speech synthesis

• Ready Player Me SDK for avatar loading and manipulation

Data Layer:

• Supabase PostgreSQL for user data, projects, job status

• Upstash Redis for caching Groq responses and job queues

• Cloudflare R2 for storing generated videos and audio files

9.3 Data Flow

1. User inputs blog URL via Next.js interface

2. FastAPI extracts content using web scraping

3. Groq API analyzes content and generates podcast script

4. Script cached in Upstash Redis for faster re-generation

5. gTTS synthesizes voice from script, outputs MP3

6. Ready Player Me avatar loaded in Three.js scene

7. FFmpeg combines audio, avatar video, and background

8. Final video uploaded to Cloudflare R2

9. Socket.io emits progress updates to frontend in real-time

10. User receives download link via WebSocket notification

9.4 Why This Tech Stack?

Cost Optimization: Entire stack is FREE with generous limits. Groq provides 14,400 free requests/day (enough for 480+ podcasts), Vercel offers unlimited frontend hosting, and HF Spaces provides 16GB RAM for backend processing.

Performance: Groq API offers blazing-fast LLM inference (<1s response time), Next.js 14 provides excellent frontend performance with server components, and Upstash Redis enables sub-millisecond caching.

Scalability: Vercel auto-scales frontend globally, Supabase handles database scaling automatically, and Cloudflare R2 provides CDN-backed storage.

Developer Experience: Next.js 14 with TypeScript ensures type safety, FastAPI provides automatic API documentation, and Ready Player Me offers simple avatar integration.

10. API Design

10.1 REST API Endpoints (FastAPI)

Endpoint	Method	Auth	Purpose	Response
/api/auth/register	POST	None	Create user account	201 {userId, token}
/api/auth/login	POST	None	User login	200 {token, user}
/api/projects	POST	JWT	Create new project	201 {projectId}
/api/projects/:id	GET	JWT	Get project details	200 {project}
/api/generate	POST	JWT	Start podcast generation	202 {jobId}
/api/jobs/:id	GET	JWT	Check job status	200 {status, progress}
/api/videos/:id	GET	JWT	Get video URL	200 {url, metadata}
/api/avatars	GET	None	List available avatars	200 {avatars[]}
/health	GET	None	Health check	200 {status: "healthy"}

10.2 WebSocket Events (Socket.io)

Event	Direction	Payload
connect	Client → Server	{userId, token}
job:progress	Server → Client	{jobId, stage, progress %}
job:complete	Server → Client	{jobId, videoUrl}
job:error	Server → Client	{jobId, error}
disconnect	Client ↔ Server	None

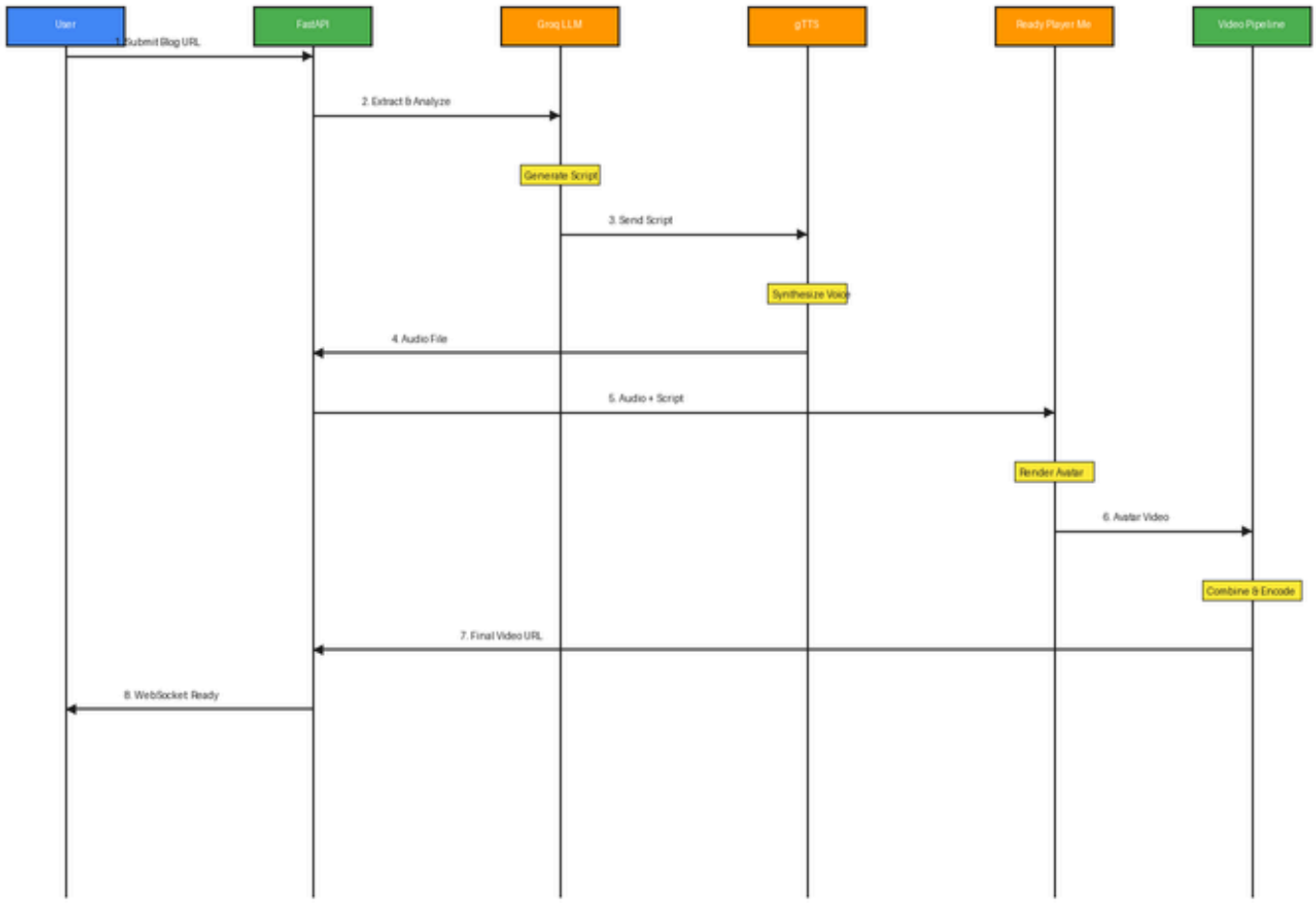
12. Technical Workflow Diagrams

The following diagrams illustrate the core technical workflows using the updated FREE tech stack:

12.1 Sequence Diagram

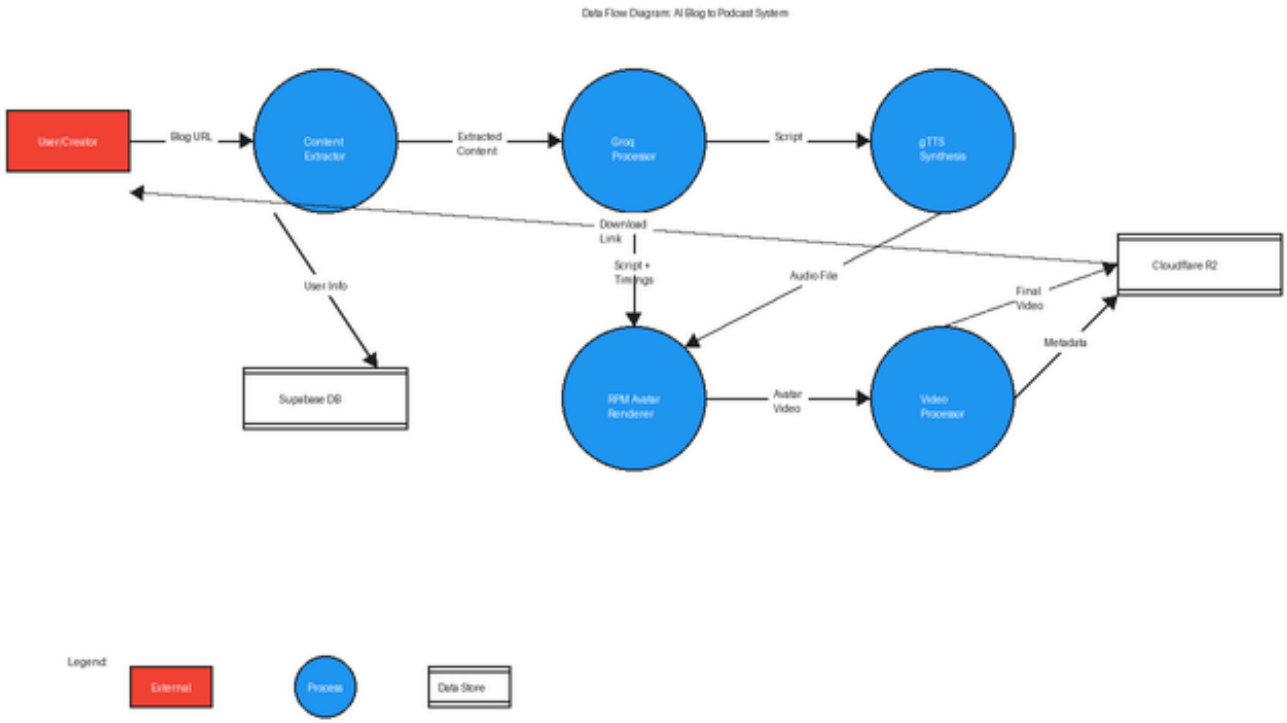
Shows the step-by-step interaction between system components during podcast generation using Groq, gTTS, and Ready Player Me:

Sequence Diagram: Blog to Podcast Conversion Flow



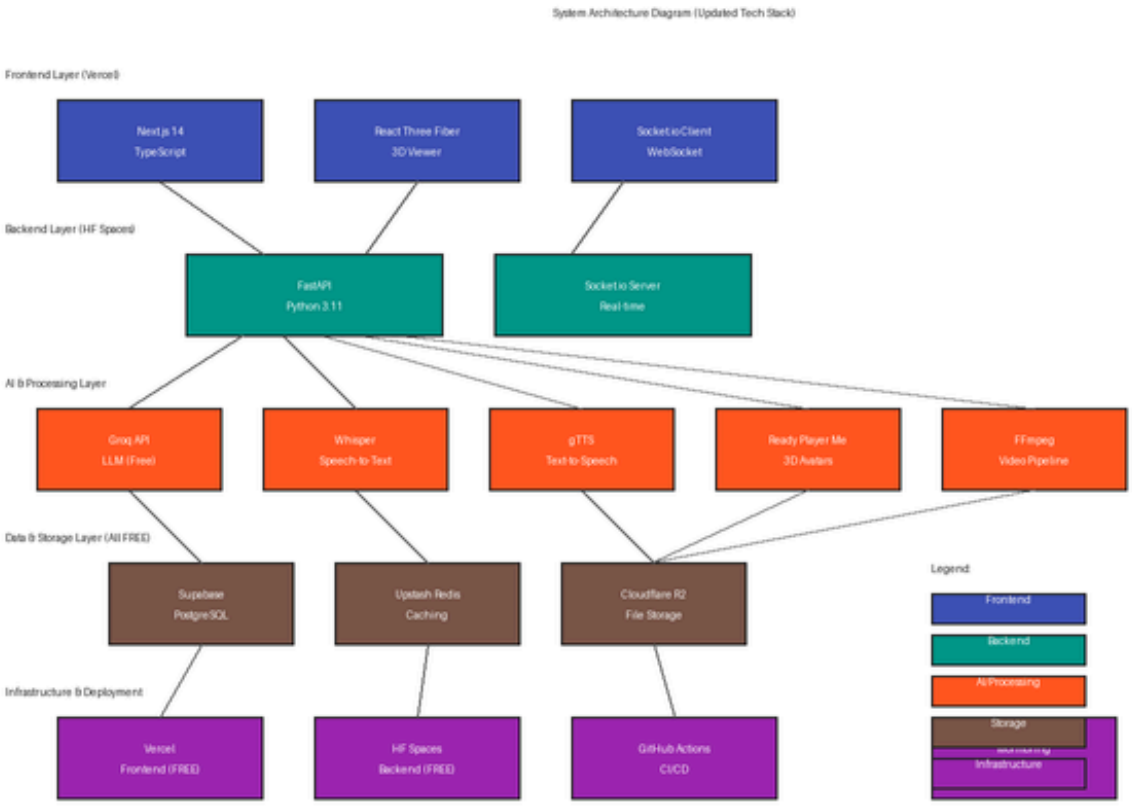
12.2 Data Flow Diagram

Illustrates how data moves through the system from user input to final video output with all FREE services:



12.3 System Architecture Diagram

Shows the complete architecture with Vercel, Hugging Face Spaces, and all FREE-tier services:



13. Quality: NFRs and Testing

13.1 Non-Functional Requirements

Metric	SLI	Target (SLO)	Measurement
Availability	Uptime %	≥ 97% (FREE tier)	Uptime monitor
API Latency	p95	≤ 500 ms	FastAPI logs
Groq Response	p95	≤ 2 seconds	API timing
Video Generation	10-min podcast	≤ 5 minutes	Job logs
WebSocket Latency	Update delay	≤ 500 ms	Client metrics

13.2 Test Plan

Area	Type	Tools	Coverage Target	Owner
Backend API	Unit	pytest	75%	Kartik
Groq Integration	Integration	pytest + Groq SDK	All endpoints	Kartik
Frontend	Component	Jest + RTL	70%	Bhaskar
E2E Flow	End-to-end	Playwright	Critical paths	Both
WebSocket	Real-time	Manual + automated	100%	Both

14. Security and Compliance

Authentication: JWT-based with Supabase Auth, secure token storage

API Security: Rate limiting (100 req/min), CORS configuration, input validation

Data Protection: Environment variables via Vercel/HF Spaces, no sensitive data in logs

Groq API: Keys stored in environment, never exposed to client

R2 Storage: Signed URLs with expiration for secure file access

WebSocket: Token-based auth, connection validation

15. Delivery and Operations

15.1 Deployment Strategy

Frontend (Vercel): GitHub integration for auto-deploy on push to main branch. Preview deployments for PRs. Automatic HTTPS, global CDN, unlimited bandwidth.

Backend (Hugging Face Spaces): Git-based deployment. Docker container with FastAPI + Uvicorn. 16GB RAM, persistent storage. Automatic SSL certificates.

Database (Supabase): Managed PostgreSQL with automatic backups. Connection pooling enabled. Row-level security policies.

Storage (Cloudflare R2): S3-compatible API. Automatic CDN caching. Public bucket for videos with signed URL expiration.

15.2 CI/CD Pipeline

GitHub Actions workflow:

1. Lint (ESLint for frontend, Flake8 for backend)

2. Type check (TypeScript, mypy)

3. Run tests (Jest, pytest)

4. Build Next.js app

5. Deploy to Vercel (automatic)

6. Deploy FastAPI to HF Spaces (on tag)

15.3 Monitoring

Vercel Analytics: Frontend performance, Web Vitals

HF Spaces Logs: Backend errors, API latency

Supabase Dashboard: Database queries, connections

Upstash Console: Redis cache hit rate

Custom logging: Groq API usage, video generation metrics

16. Risks and Mitigations

Risk	Probability	Impact	Mitigation	Owner
Groq API rate limit hit	Medium	High	Redis caching, usage monitoring, queue system	Kartik
HF Spaces downtime	Low	High	Health checks, status page, fallback error messages	Kartik
R2 storage limit (10GB)	Medium	Medium	Auto-delete old videos, compression, user limits	Both
WebSocket connection drops	Medium	Low	Auto-reconnect logic, fallback to polling	Bhaskar
Ready Player Me API issues	Low	Medium	Fallback avatars, error handling	Bhaskar
Schedule delays	Medium	High	MVP focus, weekly tracking, buffer time	Kartik

17. Research and Evaluation

Technology Research & Selection:

• Evaluated 5 LLM providers: Groq selected for speed (14,400 free req/day)

• Tested 4 TTS services: gTTS chosen for unlimited free usage

• Compared 3 avatar platforms: Ready Player Me for best FREE tier

• Analyzed deployment options: Vercel + HF Spaces for zero cost

Evaluation Criteria:

• Cost: Must be 100% FREE with no credit card required

• Limits: Generous enough for development + initial users

• Performance: Acceptable latency and processing speed

• Reliability: Good uptime and support community

Limitations:

• Groq daily limit requires usage monitoring

• gTTS voice quality lower than premium services

• HF Spaces CPU-only (no GPU for faster video rendering)

• Storage limited to 10GB (need cleanup strategy)

18. Appendices

18.1 Glossary

• Groq: Fast LLM inference platform with generous free tier

• gTTS: Google Text-to-Speech, free unlimited synthesis

• Ready Player Me: Platform for creating 3D avatars

• R3F: React Three Fiber - React renderer for Three.js

• HF Spaces: Hugging Face Spaces - free ML model hosting

• WebSocket: Bidirectional real-time communication protocol

• ASGI: Asynchronous Server Gateway Interface

18.2 Free Tier Limits Summary

Service	Free Tier Limit	Reset Period
Groq API	14,400 requests	Daily
Vercel	Unlimited deployments	N/A
HF Spaces	16GB RAM, CPU	N/A
Supabase	500MB database	N/A
Upstash Redis	10,000 requests	Daily
Cloudflare R2	10GB storage	N/A
Ready Player Me	Unlimited avatars	N/A

18.3 References

- Groq Documentation: https://console.groq.com/docs
- Next.js 14 Docs: https://nextjs.org/docs
- FastAPI Guide: https://fastapi.tiangolo.com
- React Three Fiber: https://docs.pmnd.rs/react-three-fiber
- Ready Player Me SDK: https://docs.readyplayer.me
- Socket.io Docs: https://socket.io/docs
- Supabase: https://supabase.com/docs
- Cloudflare R2: https://developers.cloudflare.com/r2

18.4 Future Enhancements (v2.0)

- Multi-language support (Spanish, French, German)
- Custom avatar creation within the app
- Batch processing for multiple blogs
- Advanced voice customization (pitch, speed, emotion)
- Background music library integration

- Direct publishing to YouTube, Spotify, Apple Podcasts
- Team collaboration features
- Analytics dashboard with detailed metrics