TrendsByAI - Technical Documentation

Architecture Overview

TrendsByAI is a modern, serverless web application designed to track, analyze, and visualize viral video trends using Artificial Intelligence.

The architecture follows a **Headless/Decoupled** pattern:

1. **Frontend**: A generic React Single Page Application (SPA) that communicates directly with the backend via APIs. It is build-agnostic and can be hosted on any static file server (Shared Hosting, Apache/Nginx).
2. **Backend (BaaS)**: Built entirely on **Supabase**. It utilizes PostgreSQL for data storage, Row Level Security (RLS) for authorization, and Edge Functions (Deno) for business logic and external API interactions.
3. **Automation Engine**: A robust cron-job system running within the database triggers Edge Functions to perform scheduled scraping, analysis, and email notifications without requiring a dedicated server.
4. **Data Source**: Integrates with the YouTube Data API v3 to fetch video metrics, facilitated by a rotating API key management system.

Tech Stack

Frontend

- **Framework**: React 18
- **Build Tool**: Vite
- **Language**: TypeScript
- **Styling**: Tailwind CSS, clsx, tailwind-merge
- **State/Data Fetching**: Supabase JS Client, React Query (inferred)
- **Visualization**: Recharts (inferred for analytics charts)
- **Icons**: Lucide React / Custom SVG icons

Backend (Supabase)

- **Database**: PostgreSQL

- **Serverless Logic**: Supabase Edge Functions (Deno / TypeScript)
- **Scheduling**: `pg_cron` extension
- **Storage**: Supabase Storage (for assets/images)
- **Authentication**: Supabase Auth

Automation & DevOps

- **Scripts**: PowerShell (`.ps1`) for automated deployment of functions, database schemas, and cron jobs.
- **Linting**: ESLint

Project Structure

```
TrendsByAI/
├── src/                        # Main React Application
│   ├── components/             # Reusable UI components (Dashboard, Headers, Card
│   ├── contexts/               # React Contexts (AuthContext)
│   ├── hooks/                  # Custom hooks (useData, use-mobile)
│   ├── lib/                    # Supabase client configuration
│   ├── pages/                  # Page views (Blog, BlogPost)
│   └── styles/                 # CSS and Tailwind imports
├── supabase/                   # Backend Configuration
│   ├── functions/              # Deno Edge Functions (Business Logic)
│   │   ├── trendai-search/     # Core search logic
│   │   ├── youtube-scanner/    # Automated scanner
│   │   └── ... (20+ functions)
│   ├── migrations/             # SQL Migration files
│   ├── tables/                 # Individual table SQL definitions
│   └── cron_jobs/              # JSON definitions for scheduled tasks
├── packages/                   # Monorepo packages
│   └── react-supabase-ssg-blog # Blog subsystem
├── deploy-all.ps1              # MASTER DEPLOYMENT SCRIPT
└── dist/                       # Production build output (after running build)
```

Database Schema

The database is normalized and optimized for time-series tracking of video metrics.

Core Tables

- `videos`: The central table storing unique video metadata (Title, Thumbnail, URL, Publish Date).
- `video_scans`: Stores raw scan data. Contains `viral_score`, `view_velocity`, and `trending_rank`.
- `video_metrics_history`: Time-series data tracking views/likes over time to calculate velocity.
- `viral_trends`: aggregates high-performing videos identified as "Viral".

System & Configuration

- `api_usage` / `api_quotas`: Tracks YouTube API quota consumption to prevent