Smart Video Thumbnail Picker

A full-stack application that allows users to upload videos to Cloudinary and dynamically pick the perfect thumbnail—either by uploading a custom image or scrubbing through the video to select a specific frame.

Key Features

- **Direct Video Uploads**: Seamlessly upload videos to a Cloudinary DAM account using the Cloudinary Upload Widget.
- **Chunked Uploads**: Automatically handles large files by uploading them in chunks for reliability.
- **Dynamic Frame Scrubbing**: An interactive slider allows for precise frame selection from the entire duration of the video.
- **Live Thumbnail Preview**: Instantly see a preview of the thumbnail as you scrub through the video.
- **Multi-Format Downloads**: Download the generated thumbnail in various formats (JPG, PNG, WEBP) on the fly.
- **Persistent Storage**: Video metadata, including duration and the selected thumbnail URL, is stored in a serverless Postgres database (Neon).
- **Modern Tech Stack**: Built with the latest features of Next.js, including Server Actions and the App Router.
- **Responsive UI**: A clean and responsive interface built with Tailwind CSS and Shadcn/UI.

Tech Stack

- **Framework**: Next.js (App Router)
- **Media Management**: Cloudinary (next-cloudinary)
- **Database**: Neon (Serverless Postgres)
- **ORM**: Prisma
- **UI Components**: Shadcn/UI
- **Styling**: Tailwind CSS
- **Deployment**: Vercel

Getting Started

Follow these instructions to get a local copy up and running for development and testing purposes.

Prerequisites

- Node.js (v18 or later)

- npm or yarn
- A Cloudinary account
- A Neon account

Installation

1. **Clone the repository:**
   git clone https://github.com/musebe/smart-video-thumbnail-picker.git
   cd smart-video-thumbnail-picker

2. **Install dependencies:**
   npm install

3. Set up environment variables:
   Create a .env.local file in the root of the project and add the following variables.
   # Neon Database Connection String (get from your Neon dashboard)
   DATABASE_URL="YOUR_NEON_POOLED_CONNECTION_STRING"

   # Cloudinary Credentials (get from your Cloudinary dashboard)
   CLOUDINARY_CLOUD_NAME="YOUR_CLOUD_NAME"
   CLOUDINARY_API_KEY="YOUR_API_KEY"
   CLOUDINARY_API_SECRET="YOUR_API_SECRET"

   # This one is exposed to the browser for client-side URL generation
   NEXT_PUBLIC_CLOUDINARY_CLOUD_NAME="YOUR_CLOUD_NAME"

4. **Set up the Cloudinary Upload Preset:**
   - In your Cloudinary dashboard, go to **Settings > Upload**.
   - Scroll to **Upload presets** and create a new one.
   - Set the **Signing mode** to **Unsigned**.
   - Name the preset smart-thumbnail-picker (or update the name in src/components/upload-zone.tsx).
5. Apply database migrations:
   This will set up the Video table in your Neon database.
   npx prisma migrate dev

6. Generate Prisma Client:
   Ensure your Prisma client is up to date with your schema.
   npx prisma generate

7. **Run the development server:**
   npm run dev

Open http://localhost:3000 in your browser to see the application.

Deployment

This application is optimized for deployment on Vercel.

1. Push your code to a GitHub repository.
2. Connect your repository to a new Vercel project.
3. Add the same environment variables from your .env.local file to the Vercel project settings.
4. Ensure your build command in Vercel is set to prisma generate && next build to keep the Prisma client in sync.
5. Vercel will automatically build and deploy your application upon every push to the main branch.