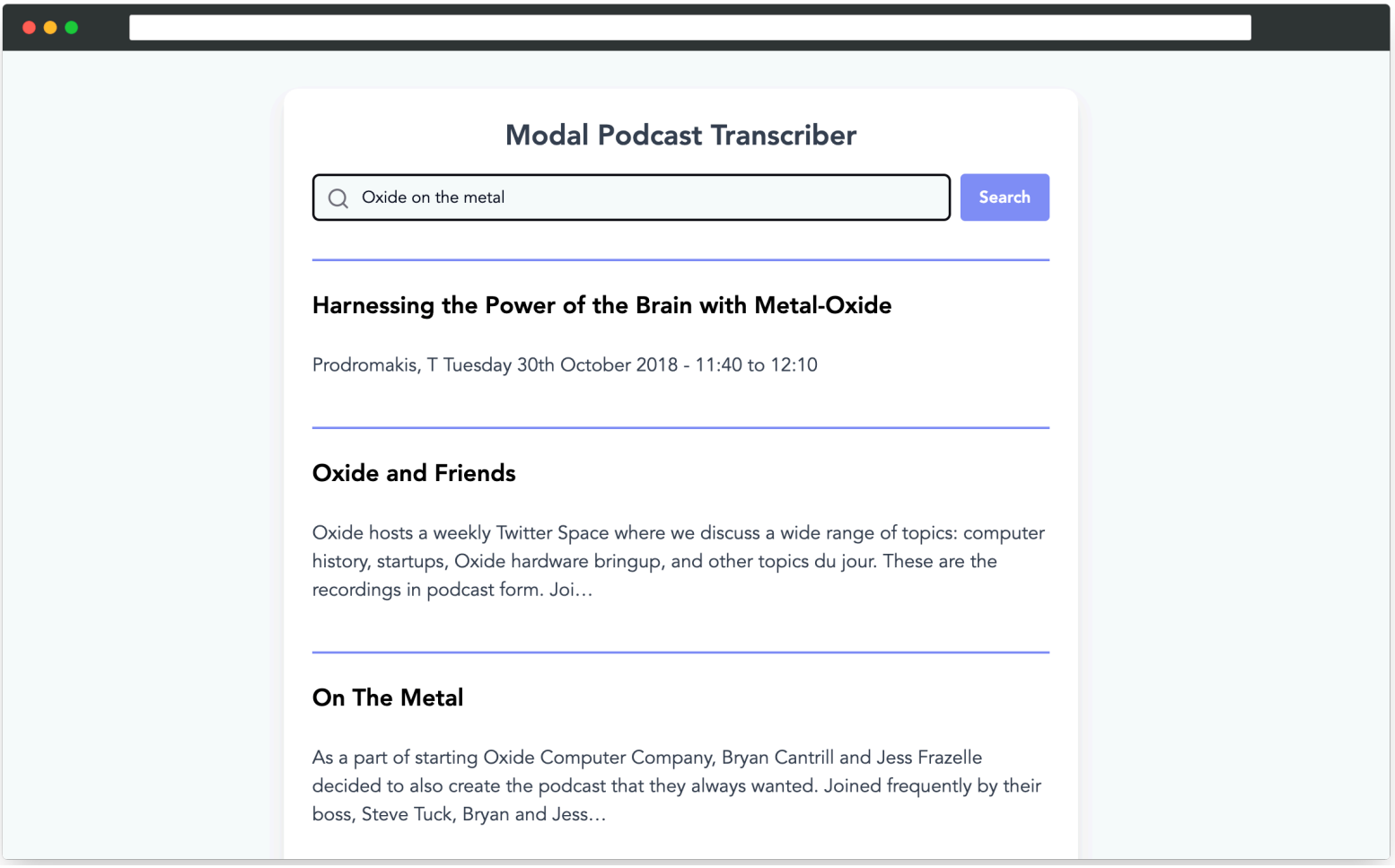
Example: Parallel podcast transcription using Whisper

OpenAI's late-September 2022 release of the *Whisper* speech recognition model was another eye-widening milestone in the rapidly improving field of deep learning, and like others we jumped to try Whisper on podcasts.

The result is the Modal Podcast Transcriber!



This example application is more feature-packed than others, and doesn't fit in a single page of code and commentary. So instead of progressing through the example's code linearly, this post provides a higher-level walkthrough of how Modal is used to do fast, on-demand podcast episode transcription for whichever podcast you'd like.

Hour-long episodes transcribed in just 1 minute

The focal point of this demonstration app is that it does serverless CPU transcription across dozens of containers at the click of a button, completing hour-long audio files in just 1 minute.

We use a podcast metadata API to allow users to transcribe an arbitrary episode from whatever niche podcast a user desires — how about *The Pen Addict*, a podcast dedicated to stationary?.

The video below shows the 45-minute long first episode of *Serial* season 2 get transcribed in 62 seconds.

0:00

What's extra cool is that each transcription segment has links back to auto-play the original audio. If you read a transcription and wonder, did they really say that? Click the timestamp on the right and find out!

(🎧 *Enable sound for this video*)

Try it yourself

If you're itching to see this in action, here are links to begin transcribing the three most popular podcasts on Spotify right now:

1. *Case 63* by Gimlet Media
2. *The Joe Rogan Experience*
3. *The Psychology of your 20s*

Tech-stack overview

The entire application is hosted serverlessly on Modal and consists of these main components:

- A React + Vite single page application (SPA) deployed as static files into a Modal web endpoint.
- A Modal web endpoint running FastAPI
- The Podchaser API provides podcast search and episode metadata retrieval. It's hooked into our code with a Modal Secret.
- A Modal async job queue, described in more detail below.

All of this is deployed with one command and costs `$0.00` when it's not transcribing podcasts or serving HTTP requests.

Speed-boosting Whisper with parallelism

Modal's dead-simple parallelism primitives are the key to doing the transcription so quickly. Even with a GPU, transcribing a full episode serially was taking around 10 minutes.

But by pulling in `ffmpeg` with a simple `.pip_install("ffmpeg-python")` addition to our Modal Image, we could exploit the natural silences of the podcast medium to partition episodes into hundreds of short segments. Each segment is transcribed by Whisper in its own container task with 2 physical CPU cores, and when all are done we stitch the segments back together with only a minimal loss in transcription quality. This approach actually accords quite well with Whisper's model architecture:

> "The Whisper architecture is a simple end-to-end approach, implemented as an encoder-decoder Transformer. Input audio is split into 30-second chunks, converted into a log-Mel spectrogram, and then passed into an encoder."
>
> —*Introducing Whisper*

Run this app on Modal

All source code for this example can be found on GitHub. The `README.md` includes instructions on setting up the frontend build and getting authenticated with the Podchaser API. Happy transcribing!