Trademark Identification & Analysis Tool

Team 11 Sprint 4 Lane Keck, Logan Taggart, Caleb Stewart

GitHub Repositories

For the application, we have now set up our two separate GitHub repos, one for the React frontend, and one for the Python backend.

By doing this, it allows for us to be able to better organize/manage our commits, and it is also essential for the automation of deployment to our hosting services.

Application Deployment

The front and back end have now both been deployed!

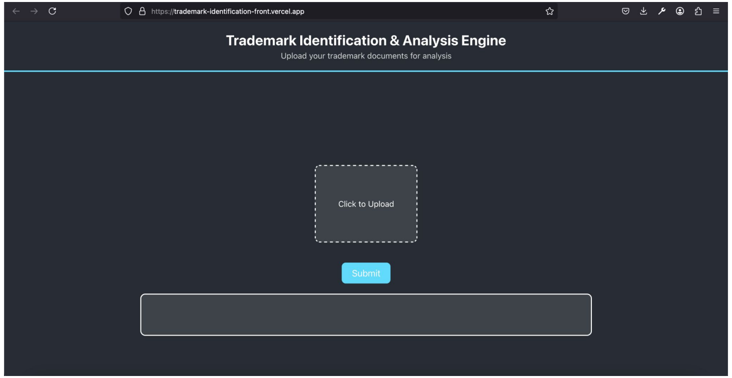
For the front we used Vercel, and for the back end we used Render (for now).

However, we have found that due to the computational heavy nature of our project, we are going to have to look into other options to be able to handle our backend.

The possible solutions will still need to be free, but they must provide us with more and better access to computing power/resources.

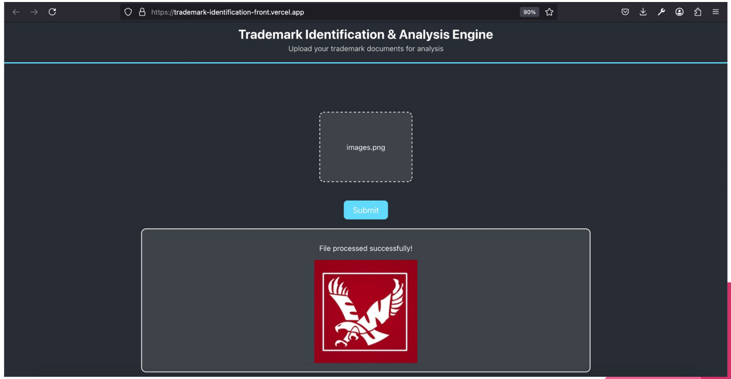
Front End Design

Below is the current design of our front end, that we've just set up as a proof of concept. Currently it takes in an image file as input, then sends it to the backend to be processed by our model before it is returned with the found logos to the user.


https://trademark-identification-front.vercel.app

Trademark Identification & Analysis Engine

Upload your trademark documents for analysis

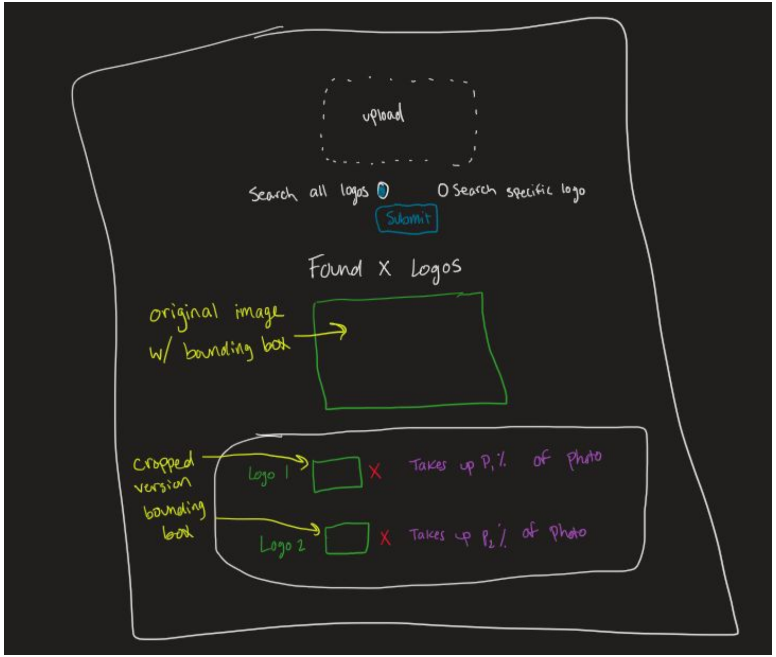

https://trademark-identification-front.vercel.app

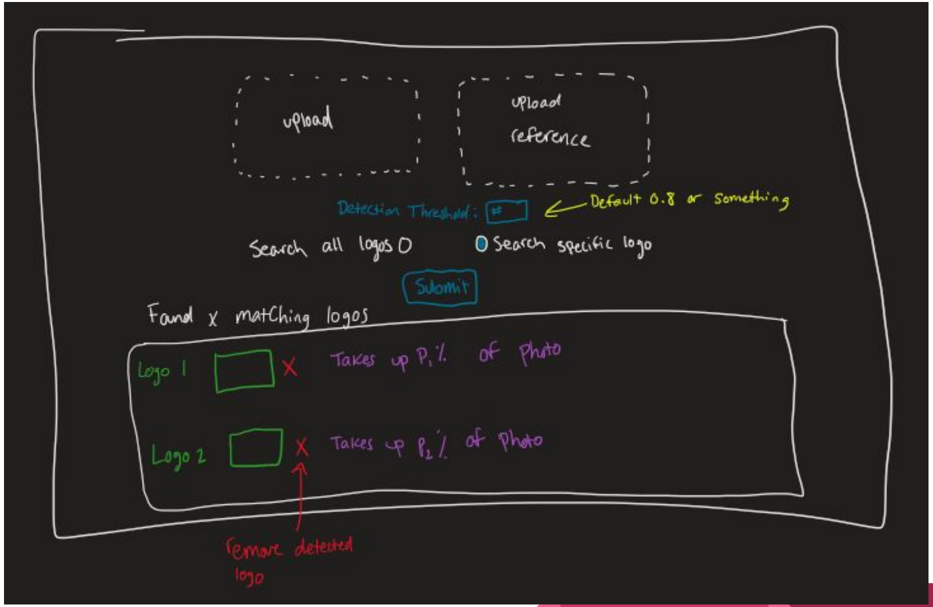
Trademark Identification & Analysis Engine

Upload your trademark documents for analysis

File processed successfully!

Planned Front End Design





Cosine Similarity

One thing that we were struggling with was how to identify a logo that was referenced, and only draw a box around that logo.

Solution: Cosine Similarity

We are able to identify similarities and differences between two vectors

Cosine Similarity

To transform our images into vectors, we can use an embedding algorithm.

Two embedding algorithms we found that gave good results:

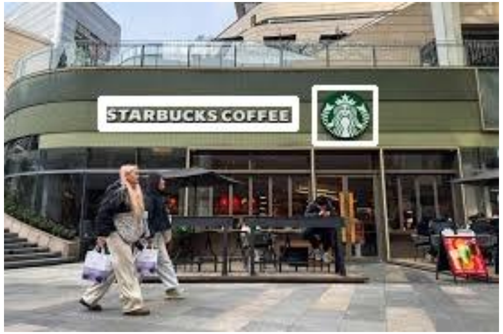
BEiT Embedding - Overall low similarity scores, but still a clear difference

CLIP Embedding - Overall high similarity scores, but still a clear difference

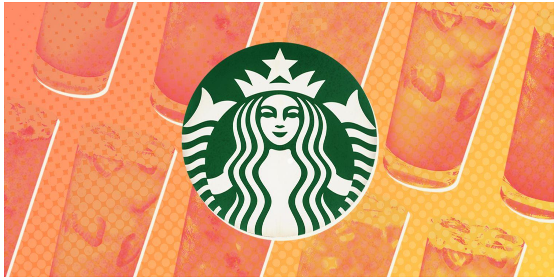
Both of these have relative success, but there may be a problem of false positives.

Cosine Similarity

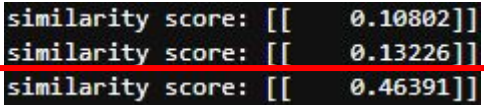
Input Image



+

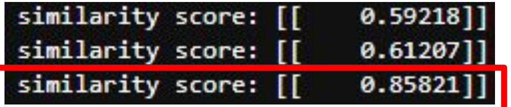


Reference Logo



```
similarity score: [[    0.10802]]
similarity score: [[    0.13226]]
similarity score: [[    0.46391]]
```

BEiT Embedding



```
similarity score: [[    0.59218]]
similarity score: [[    0.61207]]
similarity score: [[    0.85821]]
```

CLIP Embedding

Expected logo to match

Challenges

-Tracking logos in a video

-Limits on backend (slow feedback, etc)

-Front end development