Visual Algorithm Research Spike

Goal: Find a possible off the shelf visual algorithm that we could use for the purposes of facial recognition. This medium post (https://medium.com/state-of-analytics/sufficiently-advanced-technology-visual-processing-fc948503a6ba) is a great starting point for seeing the strengths and weakness of the technology as of this past December.

First Candidate: IBM - Watson Visual Recognition
Sources: Official Watson Tutorial (https://www.youtube.com/watch?v=vnWblT50w1Y), Node Github (https://github.com/watson-developer-cloud/node-sdk), Pricing and Full Documentation (https://www.ibm.com/watson/developercloud/visual-recognition.html)
Summary: Watson provides an API and node package to support the interaction with IBM's algorithm. Integration with our Node JS server would be fairly minimal (require('watson-developer-cloud/visual-recognition/v3') to speak with the API. In order to train the system, we'd need to implement Custom Image Classification and Face Detection to build up a library of detectable faces.
Pros: Great documentation, well developed node support
Cons: Pricing and feasibility are the largest ones. Watson does not provide a database of everyday normal people. In order for this to function, we must be given a sufficient enough dataset directly from social media companies, government entities, etc. and merge them all into one functional dataset. We'd then need to run all the images associated to every user and account to train the image classification. For the scope of this, we would need to build out APIs and leverage the APIs of the entities providing us the core data in order to automate the process. This cannot be done by hand.
Next Steps: Business Development would need to make contacts with these social media companies and resolve the likelihood and terms under which we might gain the data necessary to make the app function.

Second Candidate: Microsoft Azure - Face API
Sources: Product Page (https://azure.microsoft.com/en-us/services/cognitive-services/face/), SDK (https://www.npmjs.com/package/azure)
Summary: Microsoft provides a simple node package to interact with their API whose primary functions are to detect human faces and compare similar ones, organize images into groups based on similarity, and identify previously tagged people in images.
Pros: Good documentation, large SDK and enterprise support for the product offering.
Cons: Pricing again. We can test it with 30,000 image transaction for free in a month, but long term we'd need to leveraging their facial storage for as many humans in the world as possible (at $.50 per 1,000 images a month) and same logistical hurdles as Watson. We need a dataset of tagged faces to be able to offer a service that lets you search faces looking for an exact match.

Next Steps: Same as Watson. We need to know what and how we can get images paired to names from social media or government entities.

Third Candidate: White Label an existing facial recognition
Sources: Facial recognition company (http://www.cognitec.com/)
Summary: We'd work in conjunction with a company who has already put in the man hours and effort to build a database or means of scouring over third party sites to determine the identity of a given face. Possibly if we constructed our product to specifically be non-profit with the express purpose of helping those in need, we might be able to arrange use of their API with the proper contracts.
Pros: Solution is already there, we just need to skin it and negotiate access to display on our site/app.
Cons: May require significant legal and contractual hurdles to find a company arrangement for us and a contract for them that leaves all sides happy.
Next Steps: Reach out to one of these potential partners

Fourth Candidate: Facebook public pages
Sources: https://github.com/mhluska/facebook-profile-scraper
Summary:
Pros: Scrappy. Very scrappy. Github repo is a work in progress so we may need to contribute in case REST API calls from Facebook have altered since the github was last updated.
Cons: Reliability and cost to data mine. We'd need to put up a significant presence on cloud servers with resque workers or some other solution, constantly hunting facebook photos.
Next Steps: Identify if we can skip ever storing the videos ourselves and create our own script that upon finding a profile, we tag the images with their face in them with the profile's name. We then send the URLs of the facebook hosted photos directly to one of the image algorithms to train them on the face. We'll still need the cost of image processing, but that appears unavoidable.