- Notes for phone interviews -

Main projects:
The Pond, Retin.us, Tanzania

Focus on Retin.us - full stack

- Open source on github
- blog post about it
- written using sails.js+backbone
- Naive Bayes classifier
- Solved my problem when google shut down
- Aggregates - supports RSS, Atom, and RDF (old)

Tanzania - pick this if asked about technical challenge

- Mention blog post - and that I had no experience with this sort of stuff before
    - but I was determined to figure it out
- At first seemed simple
    - Started with canvas - failed quickly
        - generate 1.75mil objects (x, y, col) by scanning image
        - .1 fps (updating x,y coords each time - for animation)
    - 16ms of math+draw = 60fps
        - can't even increment a number to 1.75mil if 16ms (i.e. a for loop)
    - researched particle effects, read about how you need to animate on the GPU
        - very little concrete information on the net about what I was doing
    - for GPU I needed to use WebGL, specifically OpenGL shaders (using GLSL)
        - hard to find simple GLSL tutorials for beginners
        - it's also a huge pain to debug
    - Picked Three.js for my webgl library for its ease of use
    - Did a lot of research about custom shaders in Three.js
    - Three.js ParticleSystem - 1 mesh with vertices (vertices are the particles)
    - Still need to to generate 1.75million objects
        - really slow
        - blocks UI thread (so no loading bar)
        - uses up a ton of memory (over 1GB)
        - breaks garbage collector (if it ever tried to collect, it will crash)
            - explain that I know a lot about the js garbage collector
    - went to IRC, asked about performance, some person mentioned BufferGeometry
        - exposes raw typed arrays
            - extremely space efficient
        - 16MB memory from 1GB
    - fixed UI blocking (loading the image and creating initial map) by doing work in a WebWorker (i.e. separate thread)

Backup technical challenge - Balderdash

- designed an database associations api
- some company, can’t mention name
    - they’re building a Google Drive alternative
    - helped architect an S3 backed DB filesystem
        - taking from unix inodes
        - mark orphans - for faster lookups (avoid tree traversal)
            - came about from users sharing files