Review of React-Services

- Most common issue was a desire to share state
  - Between server and client
  - But you shouldn't
- Common to feel confused about repetitiveness

An exception: Isometric/Universal Javascript

When a server and client share JS code

- Known as "isometric" or "universal" JS
- NOT state itself
    - MAYBE code about manipulating state
    - MOSTLY validation and "generic" code
- VERY RARE for many reasons
    - Tricky to get code running in both places
        - And update at same time
    - Shared needs not common enough

Why not share state between Front/Back?

- States are similar, not identical
    - Serving different needs
    - We are running simple projects
        - Fewer differences
- Most projects aren't both sides
    - Front/Back run different places
    - Usually developed different teams
    - Often Many:1/1:Many/Many:Many
- Even when full-stack
    - Shared state = coupling
        - Changes are harder, larger

Ends serve different needs

- Server: Totality of state for one or more purposes
- Client: State for a particular user for a purpose
    - Including incomplete changes
    - Including UI state (open/closed, etc)
- "Purpose" above likely not the same

Server and Client state WILL have different models

- Similar, but not identical
- Harder to see on simple applications

Not Single Purpose

- Most Services will have multiple clients
- Most Clients will have multiple services
    - And even multiple servers of services!
- Usually developed by different teams
    - May be at different workplaces! (SaaS)

Sharing state becomes

- Effectively impossible
- Somewhat meaningless

Full-Stack on a Single Purpose Web App?

- 1 Server, 1 Client, 1 Dev/Dev Team
- Confident this won't change
    - Why such confidence?

State still couples the code bases

- Code still runs on different platforms
- Changes become painful
    - Programmers need to expect change

Harder to see difference on simple apps

Let's consider some realistic apps

Github

Here are the docs for part of GitHub REST API:

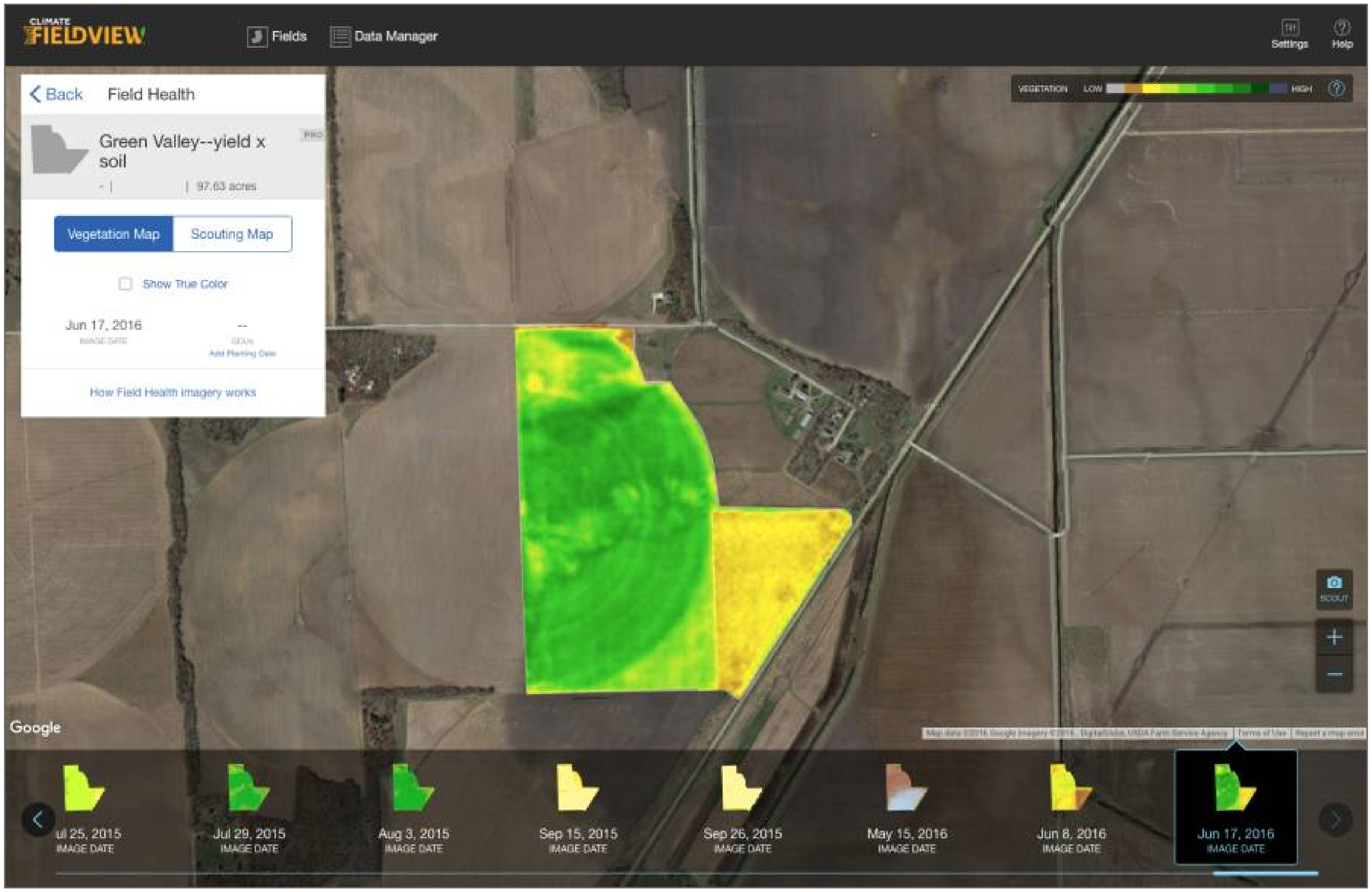
https://docs.github.com/en/rest/repos/repos

Each endpoint returns…a lot

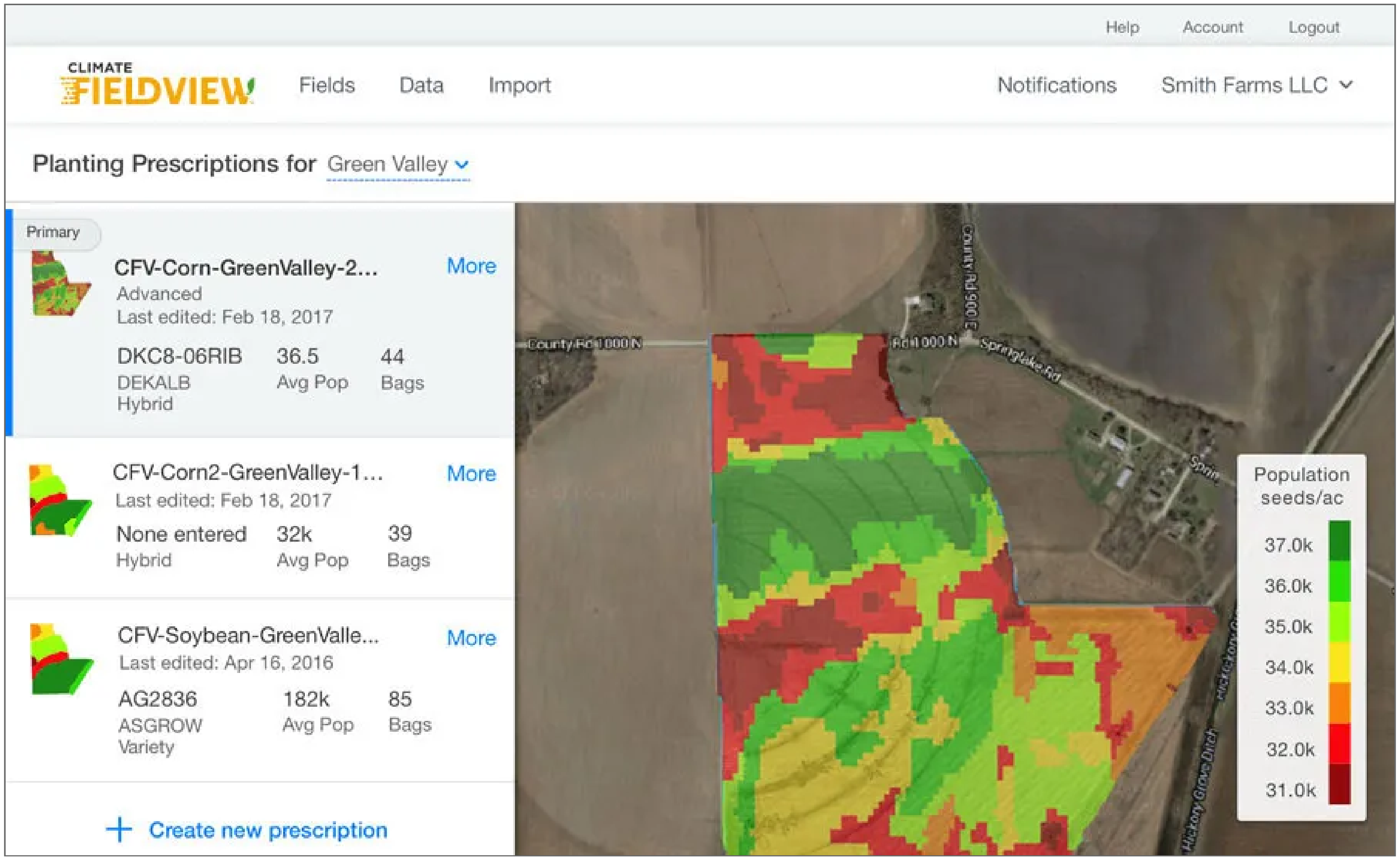
- Endpoints are CLEANED versions of state!
    - Remember our list of names from user object

This is just the Repositories section!

Climate Field Health



Climate Planting Prescriptions



Planting Prescriptions for Green Valley

Primary
CFV-Corn-GreenValley-2... More
Advanced
Last edited: Feb 18, 2017

DKC8-06RIB	36.5	44
DEKALB Hybrid	Avg Pop	Bags

CFV-Corn2-GreenValley-1... More
Last edited: Feb 18, 2017

None entered	32k	39
Hybrid	Avg Pop	Bags

CFV-Soybean-GreenValle... More
Last edited: Apr 16, 2016

AG2836	182k	85
ASGROW Variety	Avg Pop	Bags

+ Create new prescription

Climate Web App Details

- Different web apps
  - Many shared services, some different
- Notice the different headers?
  - Expect change!
- Some services
  - Image service
  - Satellite Data services
  - User service
  - Farm service
  - Field service
  - Seed Data service

Okta is a SaaS company

- Has a handful of web apps
  - Most clients are customers
- A lot of server state is describing client
  - They already know those details
  - They don't care about that
- When Okta web apps update...
  - Old web apps work for a while
  - How share state when changing models?

Summary - Client/Server are separate

- Technical Reasons
- Practical Reasons
- Change and Maintenance Reasons

When you write both

- Imagine you are two separate people