
WASHINGTON
DEPARTMENT OF NATURAL RESOURCES

Modernizing DNR's Custom GIS Viewers

WAGISA 2026 – 05-20-2026

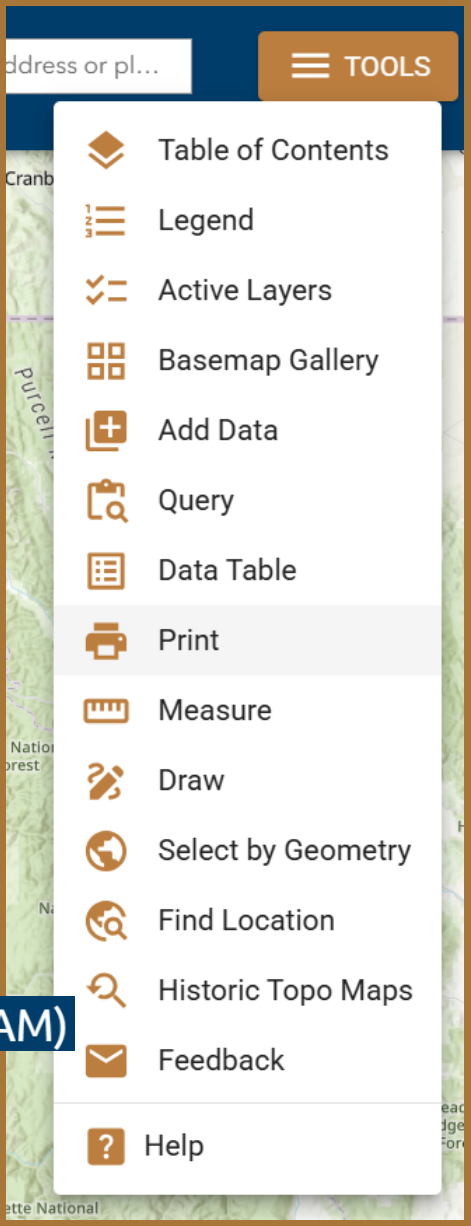
Anna Ballasiotes, GIS Application Developer
Ian Hubert, Senior GIS Developer & Project Lead
Olivia Morris, Senior GIS Developer

OVERVIEW

- Background
- Demo (not live!)
- Refactor
  - Goals
  - Decision Points
  - Process
  - Results
- Lessons Learned
- Next Steps

About the Viewers

- Custom-built Javascript web map applications
- Can be Public & Private (internal agency access only)
- Built-in tools for querying & visualizing data
- Allows users to access a variety of DNR data as well as other public data
- Divisions can customize by choosing different layers & tools



Public Viewers:

- Aquatic Resource Interactive Map (AQUARIM)
- Forest Practices Application Mapping Tool (FPAMT)
- Washington Geologic Information Portal

Internal – DNR Only Viewers:

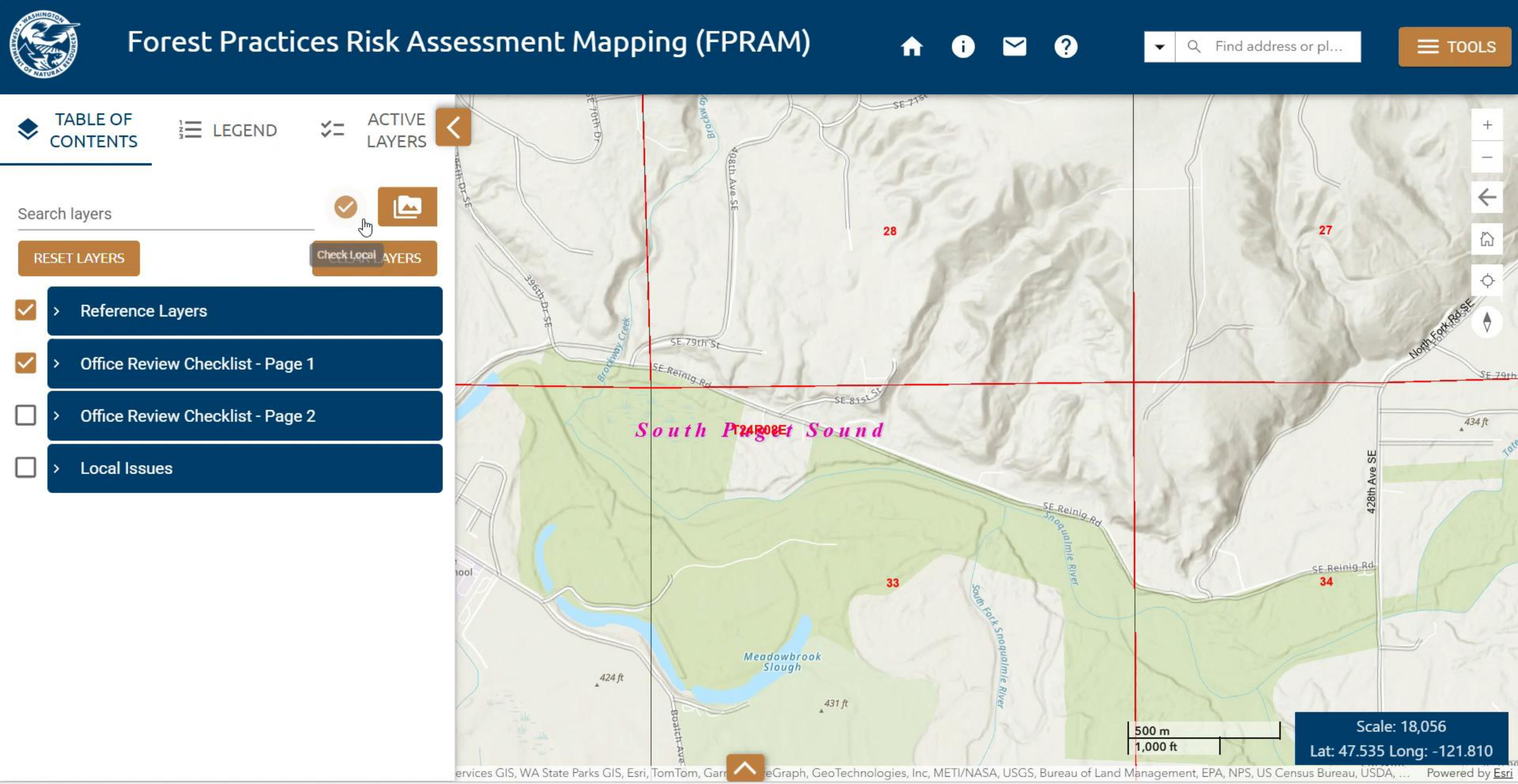
- Aquatic Resources Interactive Map(ARIM)
- Forest Practices Risk Assessment Mapping (FPRAM)
- State Uplands Viewing Tool

Business Needs

- Themed “views”
  - Quickly switch between a subset of layers turned on
- Reduce number of ArcGIS Pro licenses needed by agency
- Making DNR's map data more publicly accessible
- Custom widget requirements (printing, enhanced drawing, related tables in identify, querying, historic topographic maps)
- Giving users agency to pick a subset of layers they need out of a DNR's large stack of layers

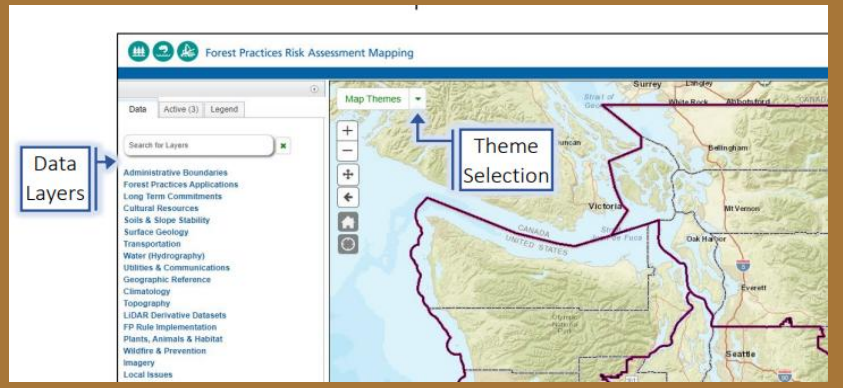
Some use cases:

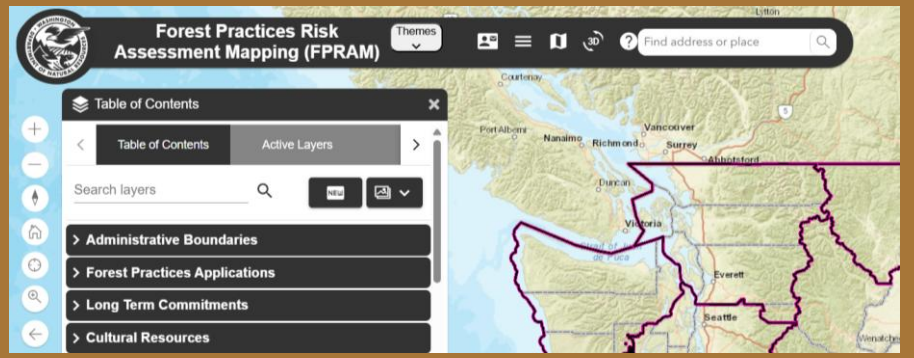
- 2,400 people in our agency → many need to look at data on a map to see coincidence, but do not need to do further analyses
- **Forest Practices** → go through checklist of items to verify an FPA
- **Geology** → reference many datasets without opening ArcGIS Pro; consultants use to easily access State data
- **SUVT** → ability to look at data & print a simple map to PDF
- **FPAMT** → small forest landowners can submit required map with FPA without needing a GIS department


Forest Practices Risk Assessment Mapping (FPRAM)
Find address or pl...
TOOLS
TABLE OF CONTENTS
LEGEND
ACTIVE LAYERS
Search layers
RESET LAYERS
Check Local
Reference Layers
Office Review Checklist - Page 1
Office Review Checklist - Page 2
Local Issues
South Puget Sound
28
27
33
34
SE 79th St
SE Reinig Rd
SE 81st St
396th Dr SE
408th Ave SE
428th Ave SE
North Fork Rd SE
Meadowbrook Slough
Snoqualmie River
South Fork Snoqualmie River
424 ft
431 ft
434 ft
500 m
1,000 ft
Scale: 18,056
Lat: 47.535 Long: -121.810
Powered by Esri

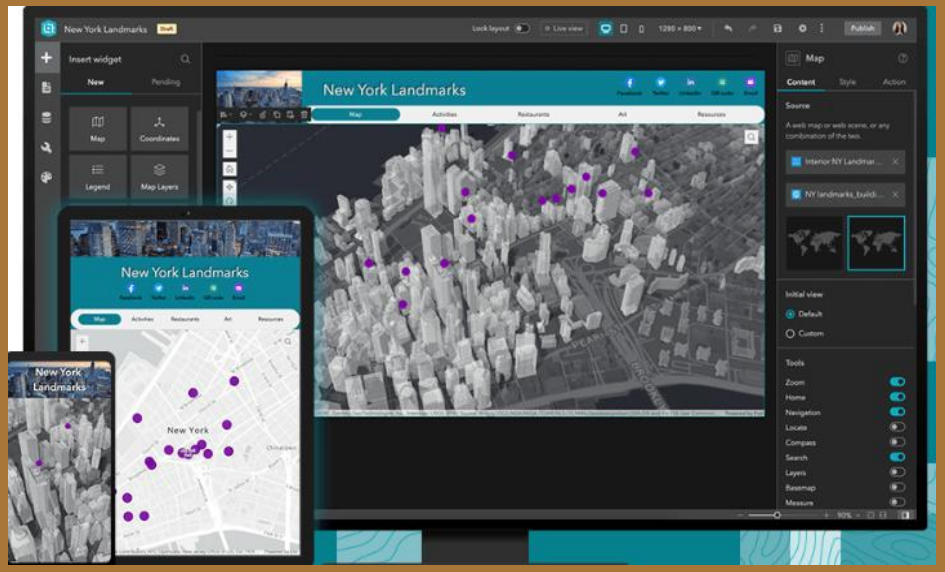
A Brief History…

- **Multiple iterations of the DNR GIS Viewers**
  - Silverlight (2008) →
  - JQuery (2015) →
  - Angular (2017/2019) →
  - **React (2024 - present)**





Why not COTS?

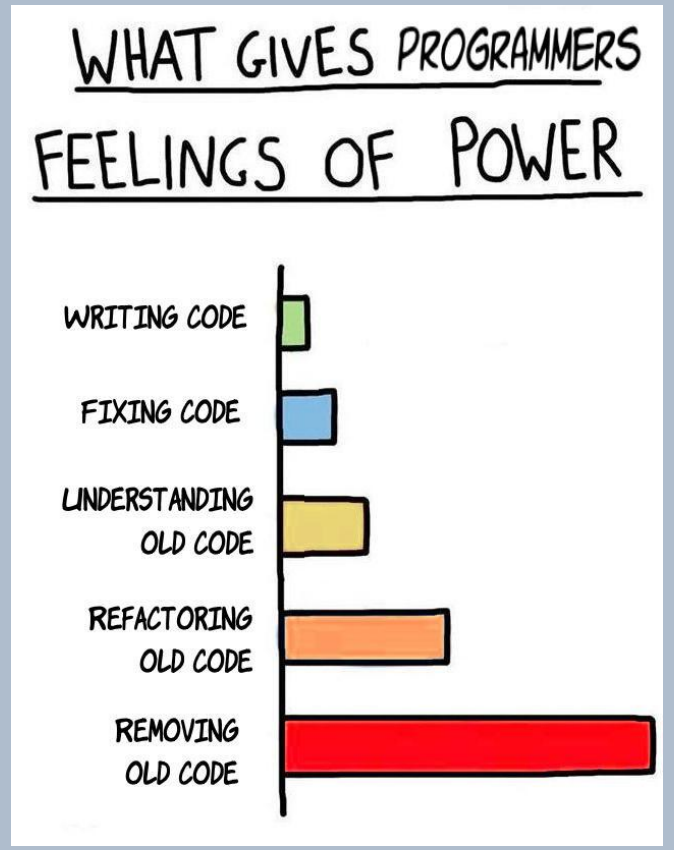


- **Esri "Map Viewer" or Experience Builder?**
  - **Volume of data** (506 layers requested!!)
  - **Data privacy & authentication**
  - **Customization to (many) division requests**
  - **Incorporate more agency branding**
  - **Would need 6 COTs unique apps vs. 1 custom app with unique configuration files**
  - **Reliant on Esri + Enterprise Versioning (& subsequent deprecation)**
    - AGOL: updates to COTS can happen at any time
    - Enterprise: updates to COTS are reliant on a wider system architecture

- **Solution: custom-built apps, more flexible to business needs**
  - Still using ArcGIS JS API/ArcGIS map components, ArcGIS Enterprise Services

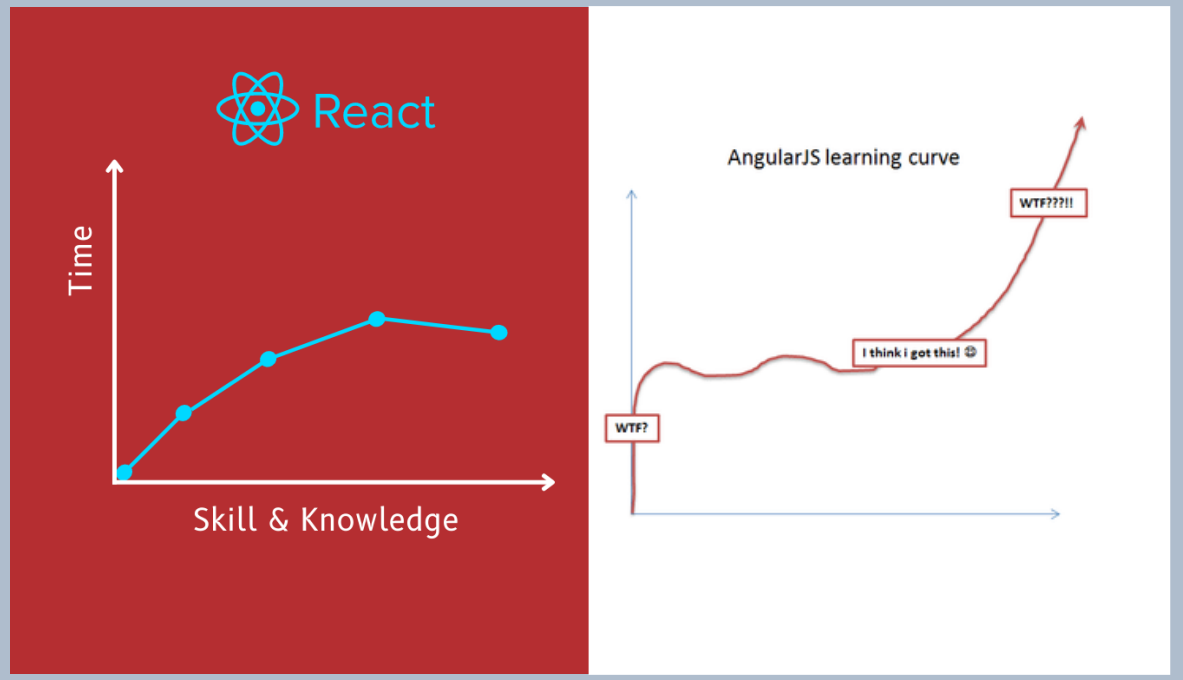
Refactor Project Objectives

- **Modernize** the application with React
- Improve **performance and maintainability**
- Training for development team
- Streamline deployment
- Enforce **better coding** practices – types, patterns, style
- Enable **business–driven layer management**
- A more **responsive UI**
- Address **feature demand** & user feedback



From Angular to React – Why?

- **Flexibility** → React offers more freedom for libraries and state management
- Easier **learning curve** → Faster development, less "abstract"
- Larger developer **community**
- Setting an agency standard for JS web applications

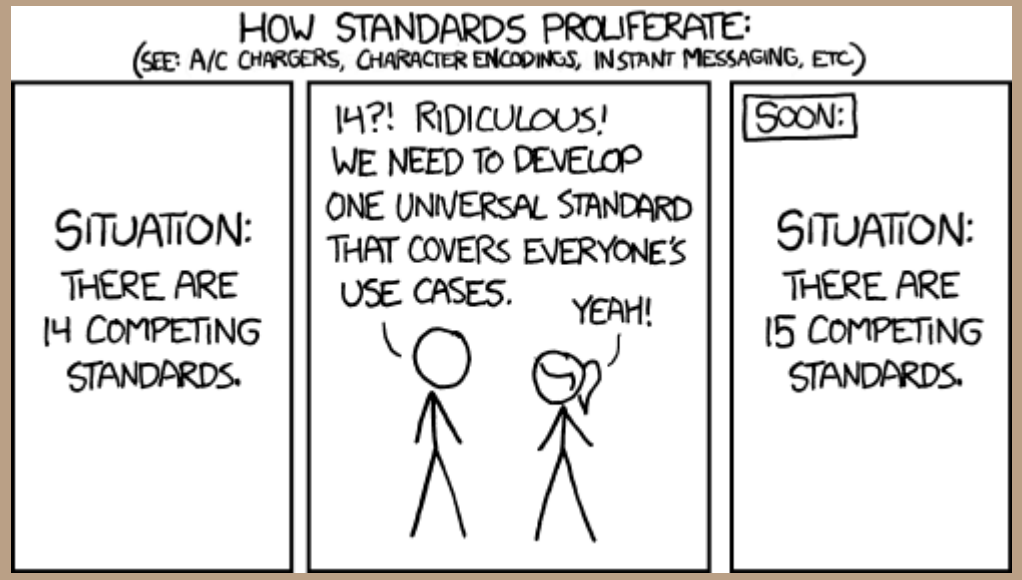


From Angular to React

```tsx
import type MapView from '@arcgis/core/views/MapView';
import '@arcgis/map-components/components/arcgis-basemap-gallery';
import { useEffect, useRef, useState } from 'react';
import mapLogic from '../Map/MapLogic';


const BasemapGalleryWidget = () => {
    const [view] = useState(mapLogic.getViewRef())
    const galleryRef = useRef<
        HTMLArcgisBasemapGalleryElement & { view?: MapView }
    >(null);

    useEffect(() => {
        if (galleryRef.current) {
            galleryRef.current.view = view;
        }
    }, [view]);

    return (
        <arcgis-basemap-gallery
            id='BasemapContainer'
            ref={galleryRef}
            style={{ height: '100%', width: 'auto' }}
        />
    );
};

export default BasemapGalleryWidget;
```

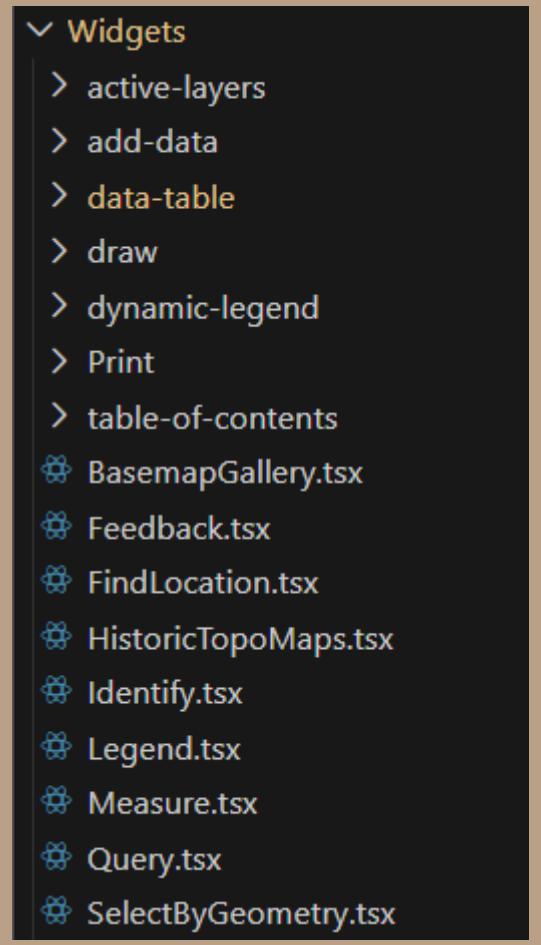
Pre-Development Process

- Project Plan submitted for approval
- Project Manager assigned
- Groundwork laid for application
  - R&D on libraries
  - Gathering Business Requirements
  - Initial app design and configuring

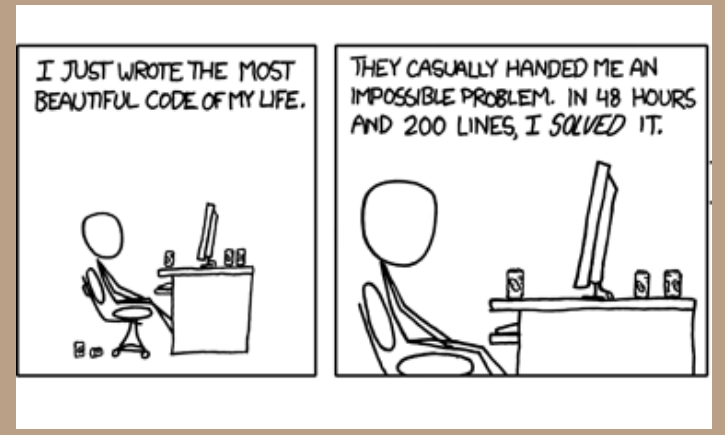


Development Process

- Developers assigned different components
- Smartsheet for Project Tracking
- Weekly project check-ins with Project Manager
- UAT from Application Owners
- Active Teams chat for regular dev communication
- Git for version control



DEV:



UAT:

... but now there's a new button and I don't like that

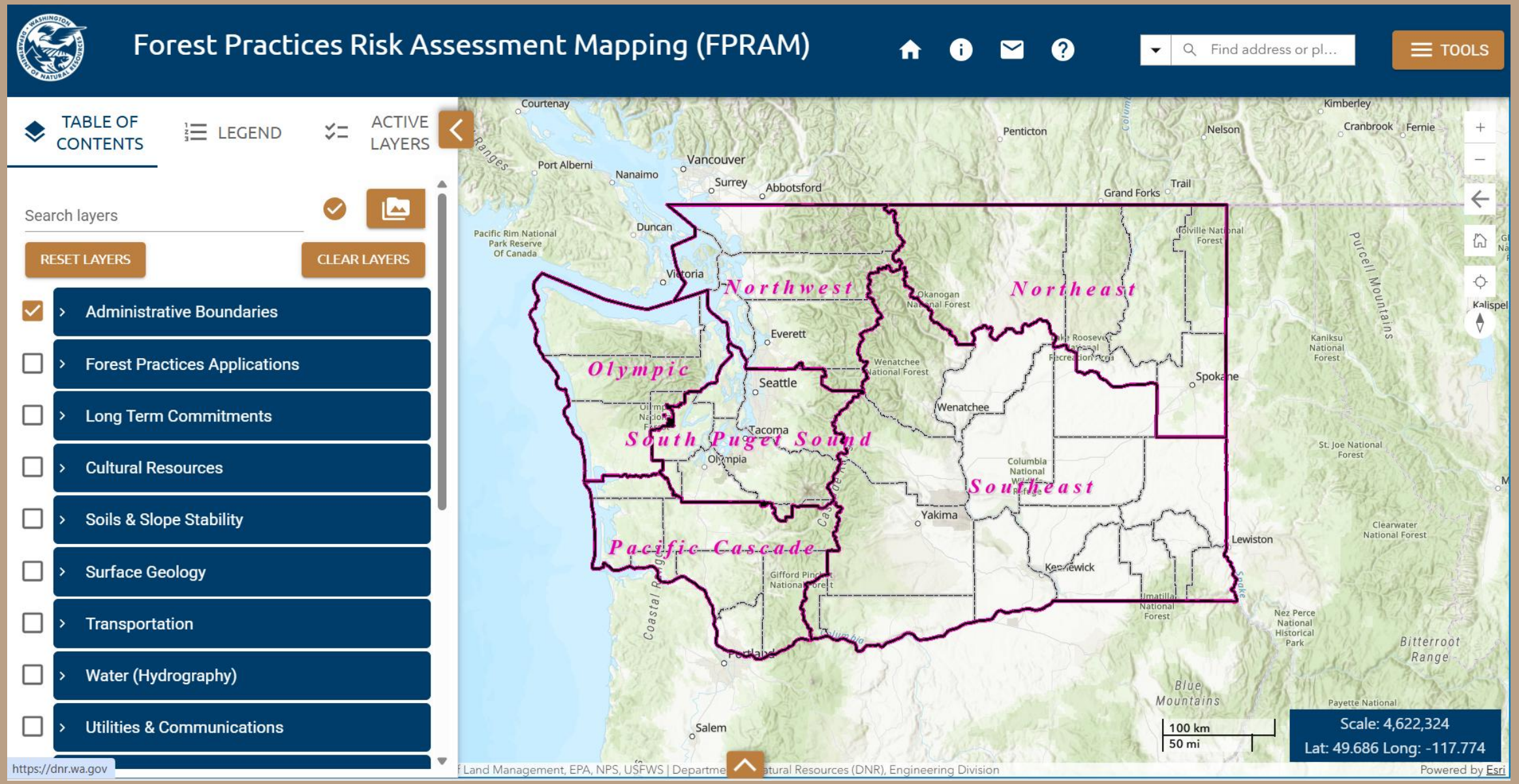
Architecture Before & After

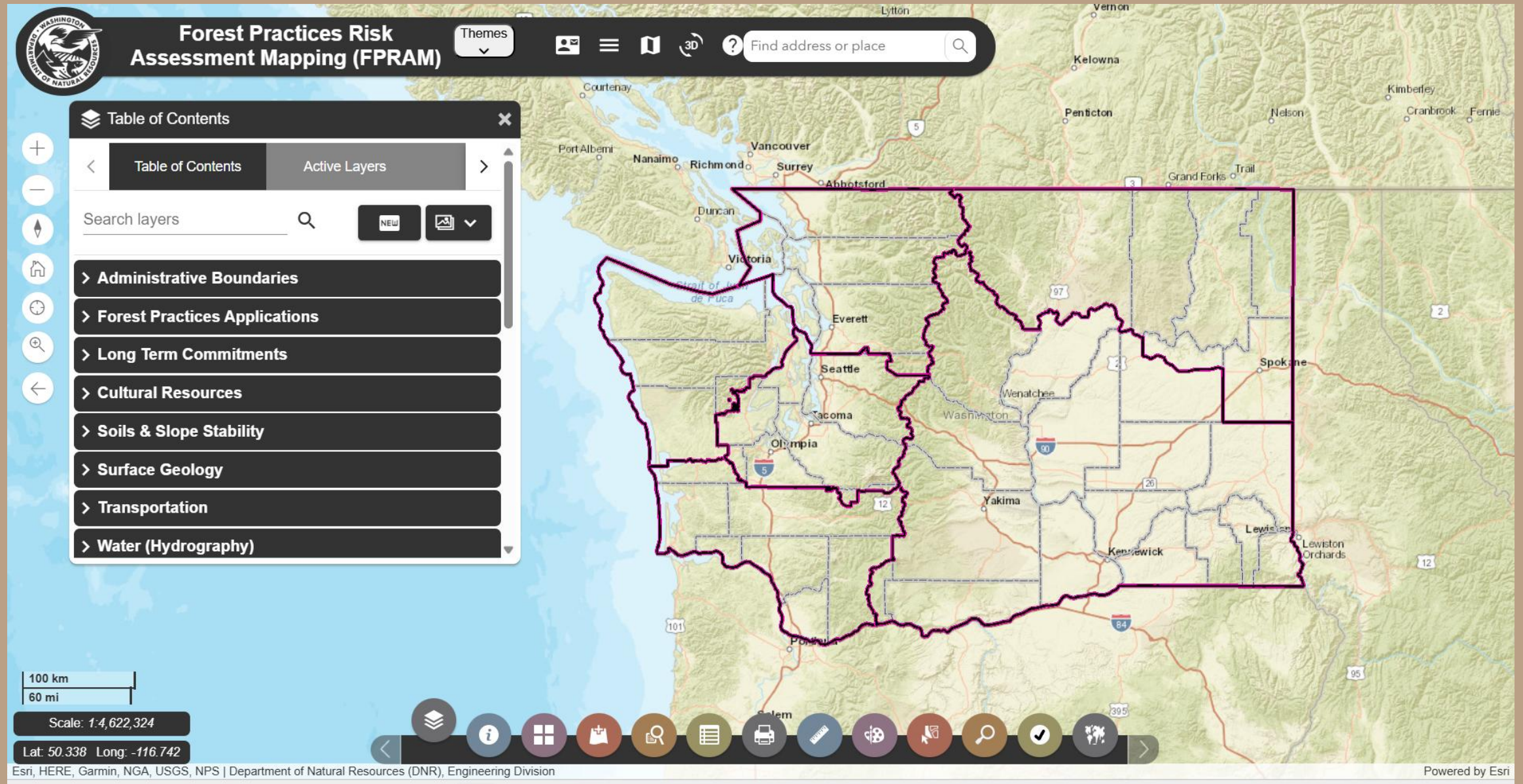
Angular App

- Deploy Time: 12 minutes no Tests
- User Load Time: 6 seconds
- Full app reload when making code changes
- Developer Resources required to add/remove layers
- Testing never worked with CI/CD
- Custom Identify Widget
- Heaps of Dead Code

React App

- Deploy Time: 7 minutes with Tests
- User Load Time: 4 seconds
- Hot-reloading using Vite when making code changes
- Business-control over adding/removing layers
- Tests using jest on CI/CD
- Use of ESRI Identify Widget
- Enforced linting & cleaned up codebase


Forest Practices Risk Assessment Mapping (FPRAM)
Find address or pl...
TOOLS
TABLE OF CONTENTS
LEGEND
ACTIVE LAYERS
Search layers
RESET LAYERS
CLEAR LAYERS
Administrative Boundaries
Forest Practices Applications
Long Term Commitments
Cultural Resources
Soils & Slope Stability
Surface Geology
Transportation
Water (Hydrography)
Utilities & Communications
Northwest
Northeast
Olympic
South Puget Sound
Southeast
Pacific Cascade
Courtenay
Port Alberni
Nanaimo
Vancouver
Surrey
Abbotsford
Penticton
Nelson
Kimberley
Cranbrook
Fernie
Grand Forks
Trail
Duncan
Victoria
Everett
Seattle
Tacoma
Olympia
Wenatchee
Yakima
Spokane
Lewiston
Kennewick
Salem
Portland
Kalispel
100 km
50 mi
Scale: 4,622,324
Lat: 49.686 Long: -117.774
Powered by Esri
https://dnr.wa.gov


Forest Practices Risk Assessment Mapping (FPRAM)
Themes
Find address or place
Table of Contents
Table of Contents
Active Layers
Search layers
NEW
Administrative Boundaries
Forest Practices Applications
Long Term Commitments
Cultural Resources
Soils & Slope Stability
Surface Geology
Transportation
Water (Hydrography)
Lytton
Vernon
Kelowna
Courtenay
Penticton
Kimberley
Nelson
Cranbrook
Fernie
Port Alberni
Nanaimo
Vancouver
Richmond
Surrey
Abbotsford
Grand Forks
Trail
Duncan
Victoria
Everett
Seattle
Spokane
Wenatchee
Tacoma
Olympia
Yakima
Lewiston
Lewiston Orchards
Kennewick
Portland
Salem
100 km
60 mi
Scale: 1:4,622,324
Lat: 50.338 Long: -116.742
Esri, HERE, Garmin, NGA, USGS, NPS | Department of Natural Resources (DNR), Engineering Division
Powered by Esri

Current Tech Stack

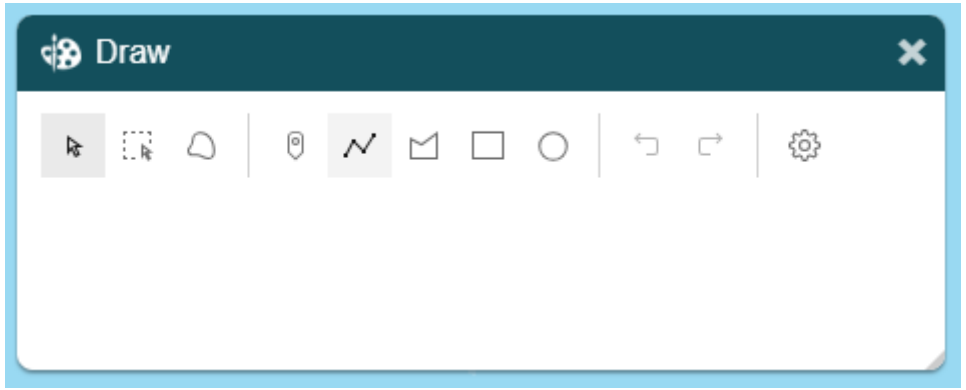
- **Git** – Version Control
- **React** – JS library
- **MUI** – Material User Interface
- **ArcGIS Enterprise Services**
- **ArcGIS Maps SDK** for Javascript
- **Vite** – JavaScript build tool / development server
- **Jest** – Testing Suite
- **Jenkins** – CI/CD pipeline

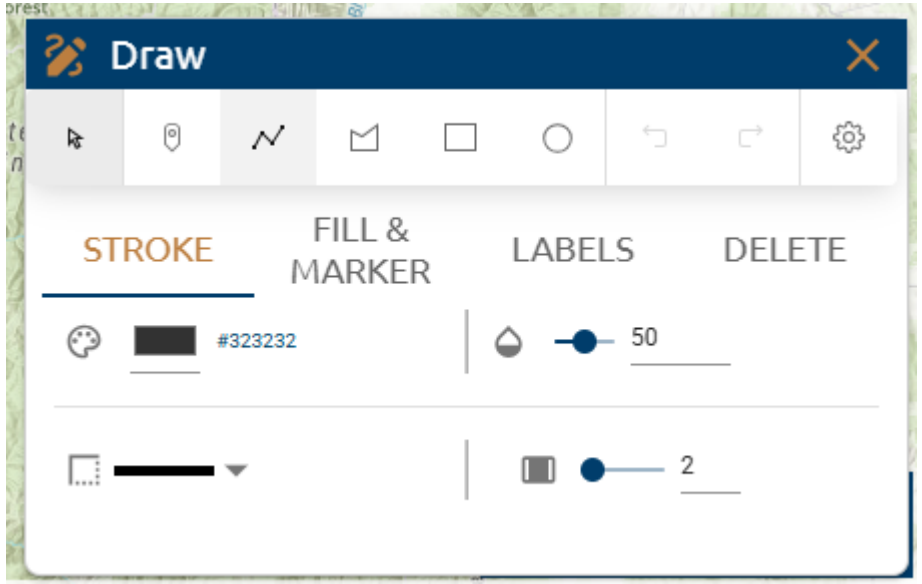
Business-Enabled Layer Management

- **ArcGIS Enterprise Hosted Table for controlling layer list in the applications**
  - No developer skills needed to edit layer list & add layers, fix broken layers
- **Editing controlled by Portal Groups**
  - Can limit control of layer list to a small group
- **App updates immediately when layer is changed & saved in table**
  - Previous update process: submit ticket -> wait for developer to fix -> dev pushes updates to app -> dev re-builds app

Draw Tool Enhancements

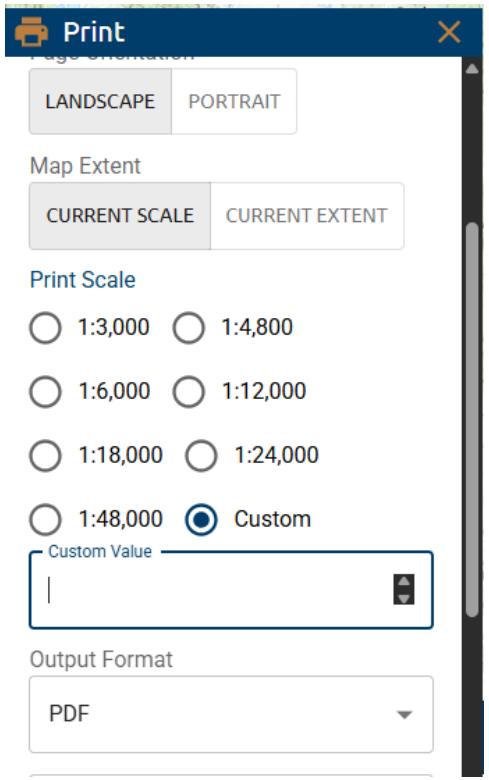
- Ability to add text labels
- Ability to modify symbol & color of points, lines, polygons, & text
- Expanded the functionality of ESRI Sketch widget

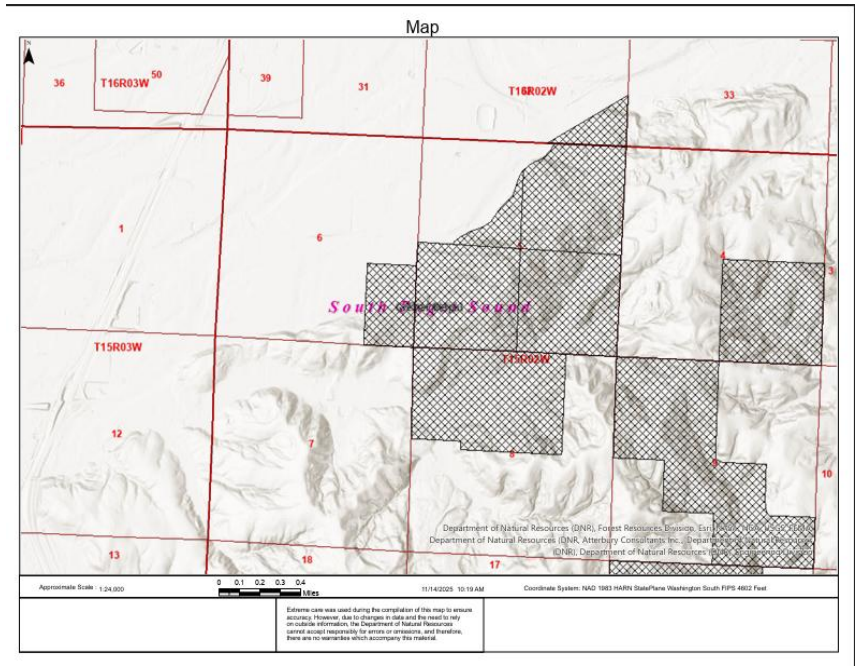




Printing Enhancements

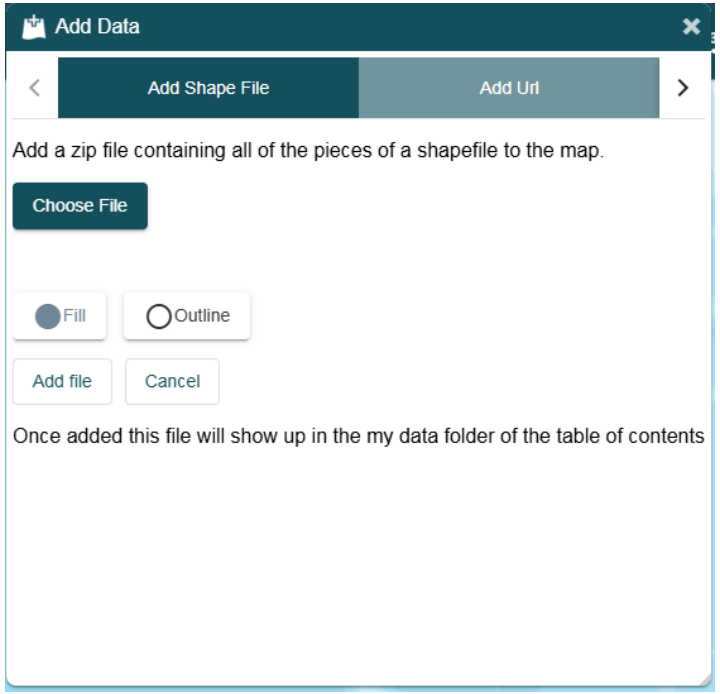
- Ability to set user-defined scale
- Additional layout sizes and orientations
- FPAMT – Multiple themed exports
- Refactor of GP Service that generates PDF maps

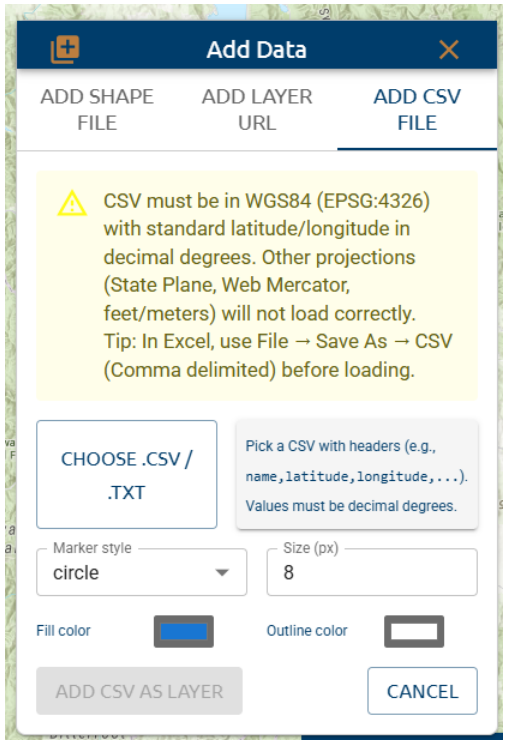




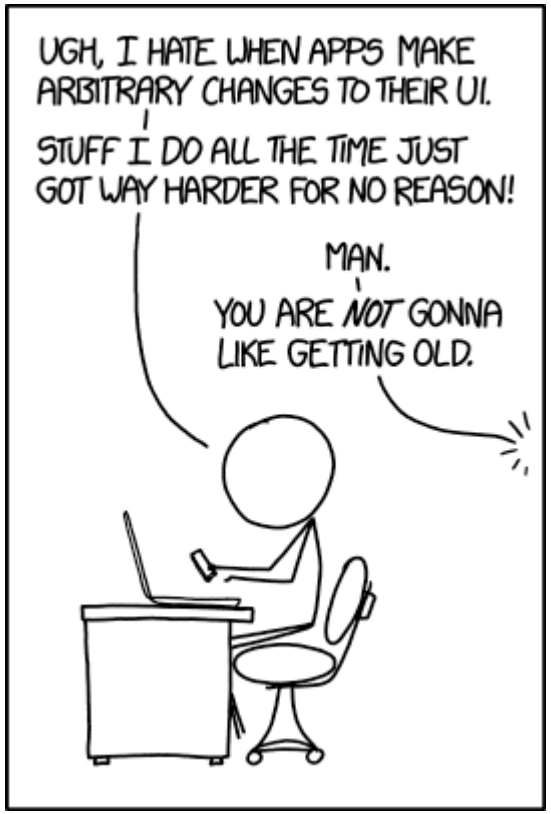
Adding CSV Files

- ‘Add Data’ Widget allows users to add shapefiles, layers from a REST URL, and CSV file





Learning the Hard Way



- 2-way binding vs 1-way binding (Angular vs React)
- React Event Handling = less straightforward
- More libraries = more decisions
- People hate change

Future Enhancements

Widgets

- Custom query functionality → Query can create new layer
- Ability to change symbology on an existing layer

UI/UX

- Better Mobile UI/UX
- Preventing widget refresh in the bottom toolbar

Backend

- Expanding testing frameworks
- Enforcing version control for incremental changes

QUESTIONS?