MonoScript

Building a versatile JS mono repository

Introduction

- My formal education background is in Geography & Geographical Information Systems.
- I moved to Columbus 7 years ago from Dayton, after graduating from Sinclair Community College to pursue my BA at Ohio State.
- In 2018, I made a pivot from GIS to CSE via a We Can Code IT Bootcamp.
- Since the graduation I've worked at several companies:
    - Nationwide
    - Smart Cosmos / Bibliotheca
    - NetJets

My Technological background

Nationwide

- Nationwide Mobile
  - Angular, Cordova, Ionic
- Nationwide Internal Resource tracker
  - Microservices / Polyrepo
  - Angular, Java, AWS, Docker, K8s
- Nationwide Cybersecurity
  - Angular, Azure, IAM

Smart Cosmos

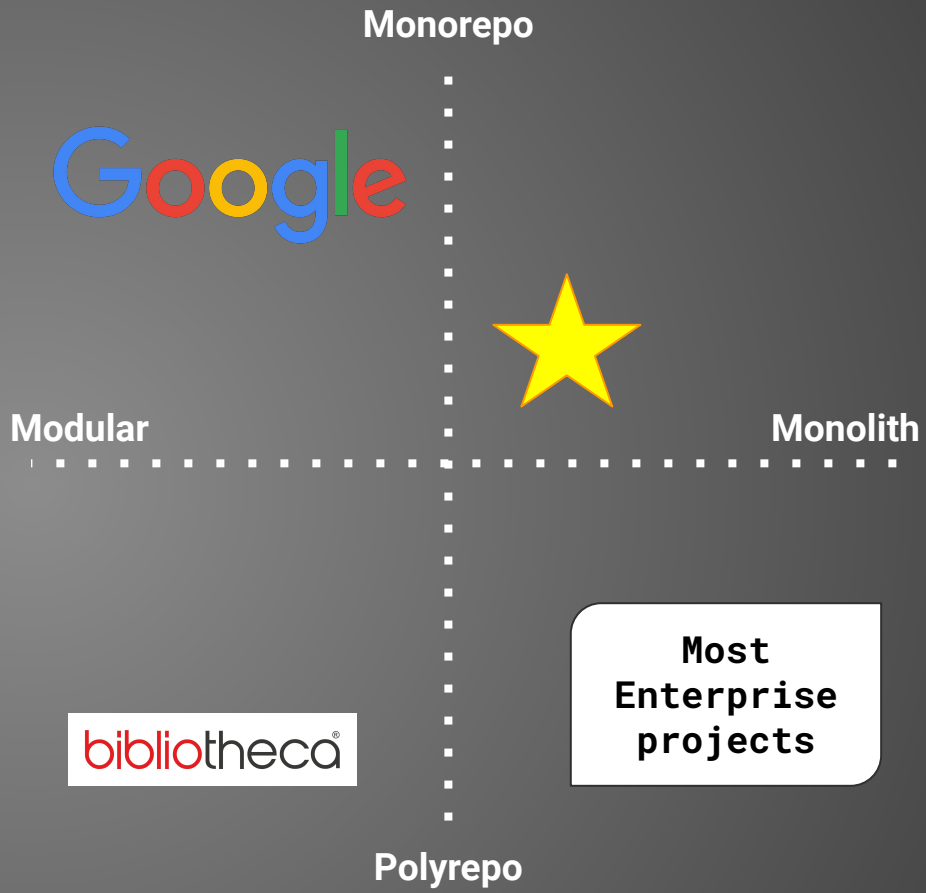
- Physical Asset Management
  - Microservices
  - React, AWS, Docker, K8's
- Cloud Library 2.0 (Bibliotheca)
  - Microservices
  - React, Java, AWS, K8's

NetJets

- Owner Portal
  - React Native web
- Owner Web
  - Monorepo
  - React, .NET, jQuery, AWS
- Design System
  - Monorepo
  - React Native, Lerna, NodeJS, CI/CD
- Design System 2
  - Monorepo
  - React Native, Turbo Repo, NodeJS, CI/CD

So what is a monorepo?

- Until joining NetJets, I believed that poly repos were the best practice for code.
- At its core a monorepo is a software strategy for containing any number of projects in a single VCS.
    - This practice dates back to the turn of the millenia thru the strategy of shared codebases.
    - Defined by having multiple distinct projects with well-defined relationships.
    - This level of organization and strategy is how we avoid a monolithic repo. A large repo lacking encapsulation and division of parts.



Poly Repos

Many repositories usually consisting of singular build artifacts. Where each team has a repo(s) for each package, app or project.

Pros:

- Team autonomy,
- Tech stack choice
- Smaller logical code repos.

Cons:

- Difficulty in code sharing
- Duplicate code
- Difficulty with shared libs and versions
- Inconsistent tooling
- Overhead in creating, configuring and connecting new projects.

This is a personal opinion

Monorepo

Numerous distinct projects in single VCS repo. With well defined organization, code sharing and project structure.

Pros:

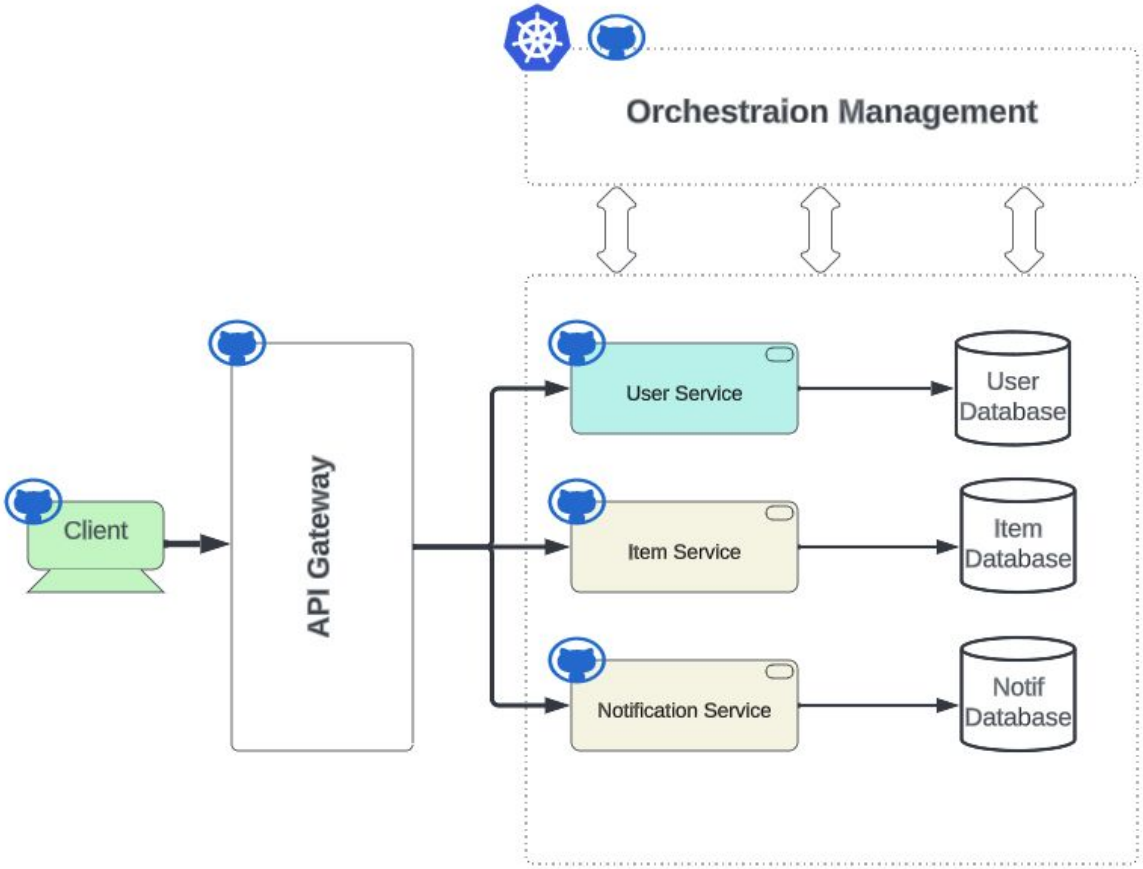
- Reduced breaking changes and easier versioning.
- Ease of setting up new projects
- Quicker on boarding
- Smaller logical code packages and apps.
- Code consistency and quality.
- Rapid access of code across teams

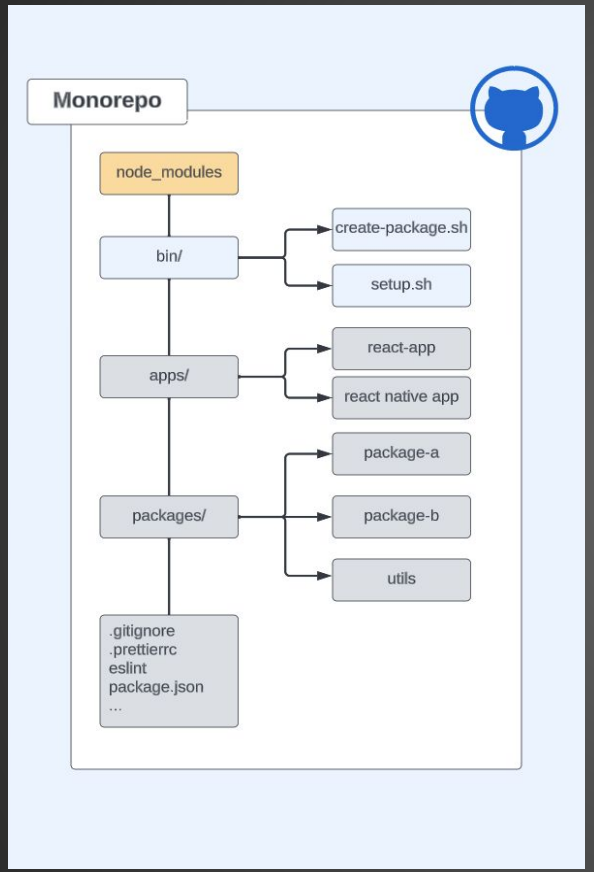
Cons:

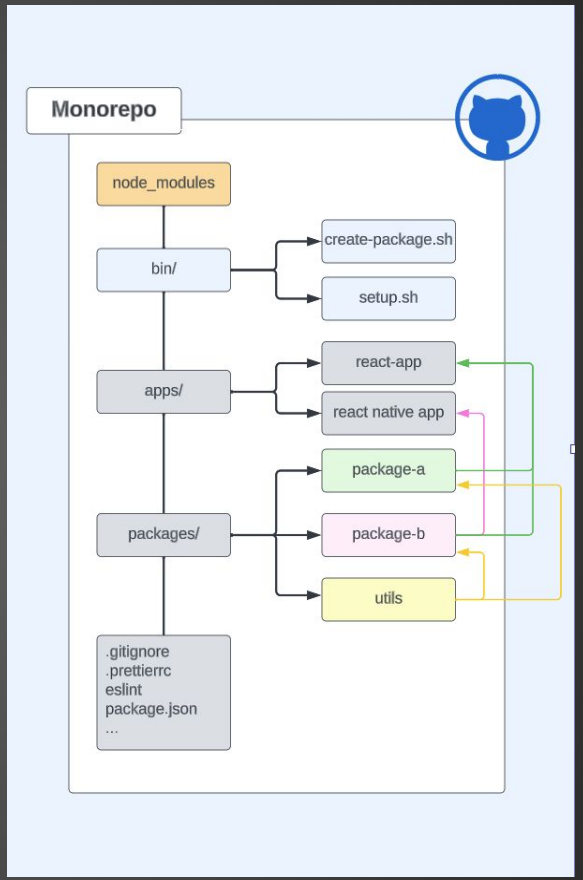
- Team autonomy
- Duplicate code
- Regulated tech stack
- Personal experience, kind of hard to deploy without the use of a manager.
- More PR emails from parts of the repo you might not be involved in

This is my personal opinion

WHICH WILL YOU CHOOSE?!?


Orchestraion Management
Client
API Gateway
User Service
Item Service
Notification Service
User Database
Item Database
Notif Database


Monorepo
node_modules
bin/
create-package.sh
setup.sh
apps/
react-app
react native app
packages/
package-a
package-b
utils
.gitignore
.prettierrc
eslint
package.json
...


Monorepo
node_modules
bin/
create-package.sh
setup.sh
apps/
react-app
react native app
packages/
package-a
package-b
utils
.gitignore
.prettierrc
eslint
package.json
...

Monorepo Managers

Turbo

Pros:

- Very easy to get started with a new repo in minutes
- Great documentation
- Scaliable
- Caching
- Parallelization of task execution
- Pick dependency manager.

Cons:

- Creates a “bloated” base project
- Pushes devs to use typescript

Lerna

Pros:

- Targeted dependency scripts
- Efficient parallelization of order execution
- Scaliable
- Workspace graphic visualization
- Caching

Cons:

- Minimal preconfiguration
- Lerna takes no responsibility for symlinking
- Limited Documentation

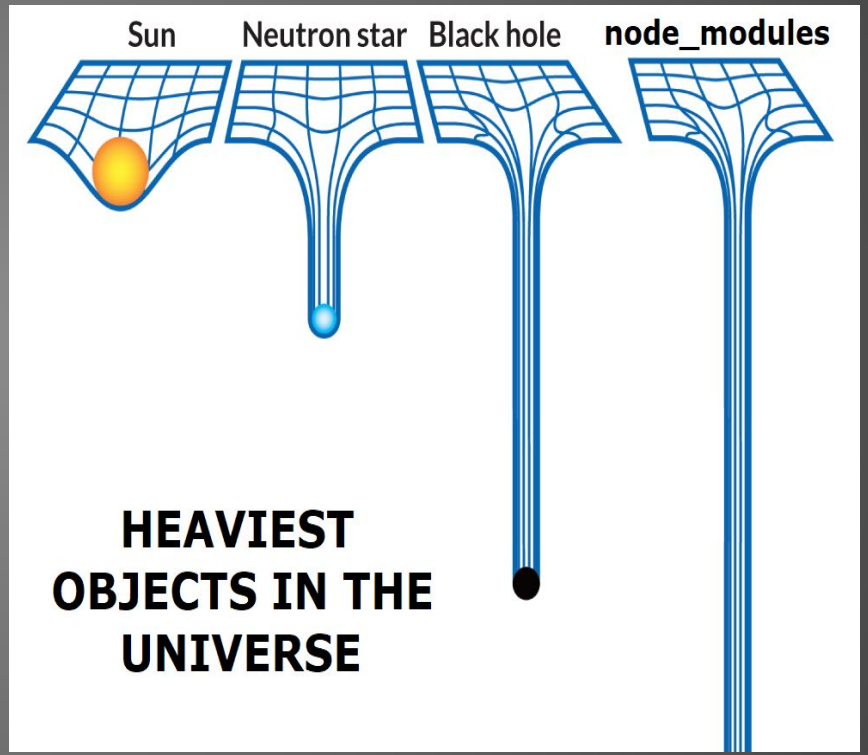
Pros for both

Dependency Hoisting

Add dependencies and devDependencies at the root of the mono, and they're accessible through out.

Have a dependency that's only used once? You can declare it at the package level package.json

Using React Native or Expo in a monorepo? You can limit hoisting to the package level.



Dependency Managers

NPM

Yarn

PNPM

This is at the discretion of the architects or developers and which fits their needs.

I know this can be a touchy subject with some JS developers.

Lerna

A quick Lerna demo

Starting a new Lerna monorepo.

This is a very basic project setup.

Lerna leaves all of the project structure setup to the developer

Developers could create scripts to quickly build out general project structure.

Utilizes dependency managers workspace config in the root package.json

Easy publishing and versioning with built in lerna CLI tools.

```
$ mkdir lerna-demo

$ cd lerna-demo

$ npx lerna init

$ mkdir packages

$ mkdir packages/utils packages/pkg-a

$ cd packages/utils && npm init

$ mkdir src

$ touch index.js src/index.js
```

Create a new blank directory

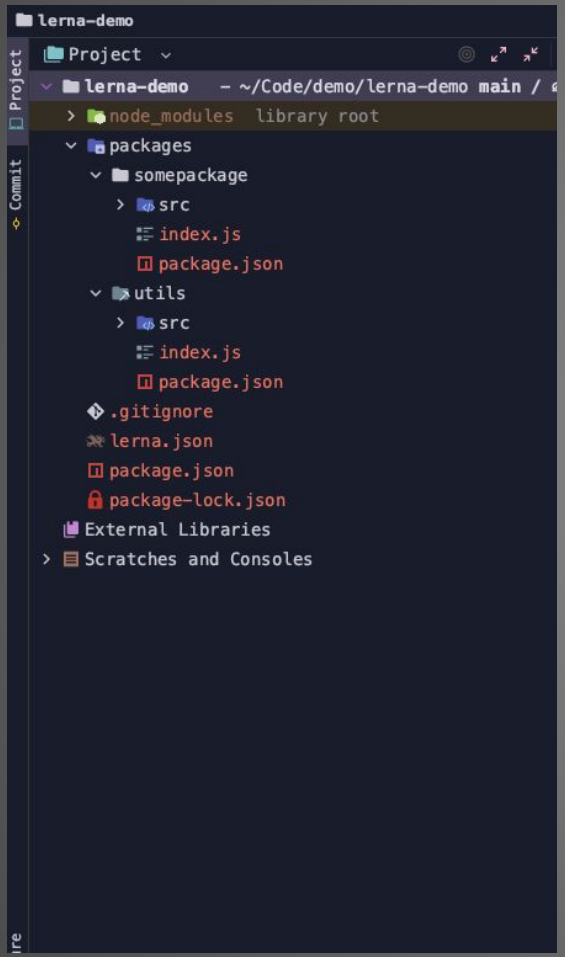
Move to that directory

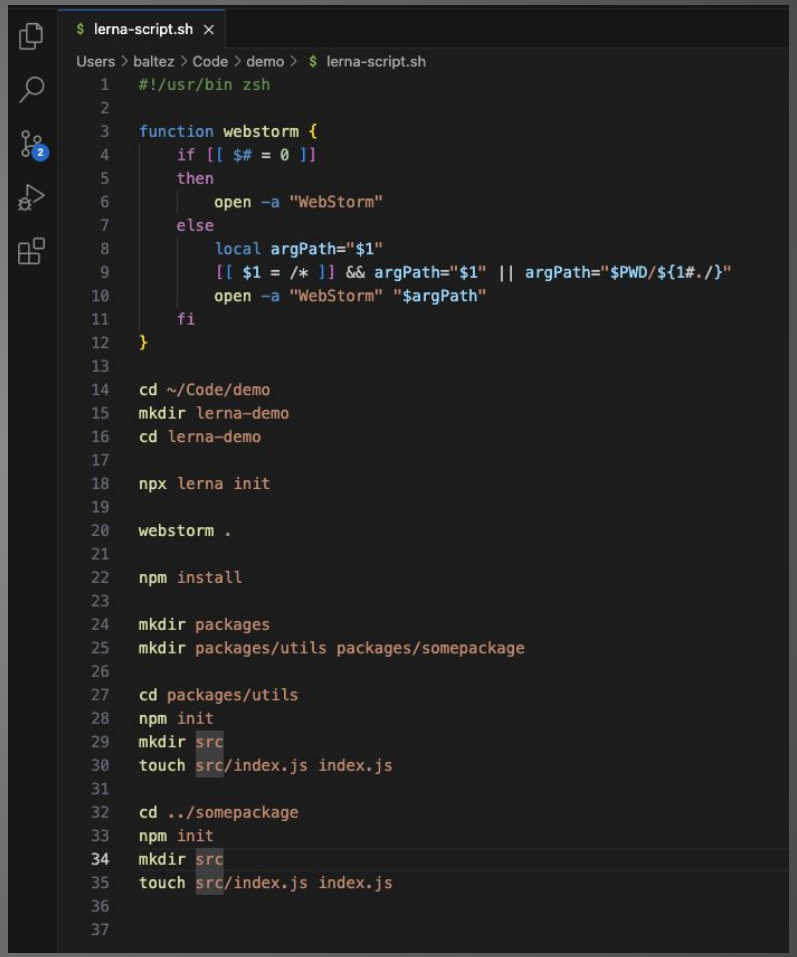
Use NPX to run lerna commands

- Generates a package.json, .gitignore & lerna.json

Now create your desired project structure.

Creating a util package that can be shared across all apps and packages as easy as adding a new dependency to dependencies or devDependencies in that projects package.json


lerna-demo
Project
lerna-demo – ~/Code/demo/lerna-demo main /
node_modules library root
packages
somepackage
src
index.js
package.json
utils
src
index.js
package.json
.gitignore
lerna.json
package.json
package-lock.json
External Libraries
Scratches and Consoles
Project
Commit


$ lerna-script.sh
Users > baltez > Code > demo > $ lerna-script.sh
#!/usr/bin zsh
function webstorm {
if [[$# = 0]]
then
open -a "WebStorm"
else
local argPath="$1"
[[$1 = /*]] && argPath="$1" || argPath="$PWD/${1#./}"
open -a "WebStorm" "$argPath"
fi
}
cd ~/Code/demo
mkdir lerna-demo
cd lerna-demo
npx lerna init
webstorm .
npm install
mkdir packages
mkdir packages/utils packages/somepackage
cd packages/utils
npm init
mkdir src
touch src/index.js index.js
cd ../somepackage
npm init
mkdir src
touch src/index.js index.js

A quick Turbo repo demo

Starting a new Turbo repo.

This is a very basic project setup.

Turbo out of the box creates a project with Typescript already configured with a base and extended for Next.js and React

apps/ with a Next.js app

packages/ with a eslint, tsconfig & UI package that are modules to keep

Feels like it forces the developer to use Typescript

Well documented and numerous demo repos for different use cases.

Easy publishing and versioning with built in lerna CLI tools.

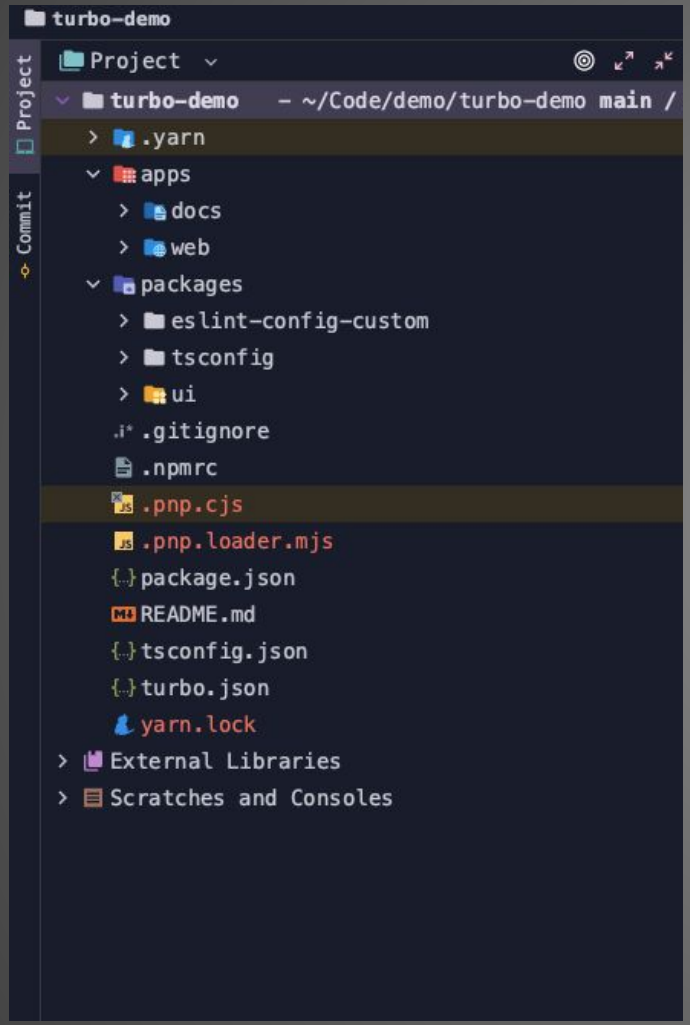
```
$ cd code/

$ yarn dlx
create-turbo@latest

$ cd turbo-demo

$ webstorm .
```

Move to the desired directory

Use Yarn dlx to run turbo commands

- Generates a package.json, .gitignore, turbo.json, packages/, apps/, tsconfig, and more.

The project is ready to go. All that needs to be done is add additional packages
If using typescript add a tsconfig.json to the new package and add the dependency package.




```
baltez@Johns-MacBook-Pro ~/Code/demo npx create-turbo@latest

>>> TURBOREPO

>>> Welcome to Turborepo! Let's get you set up with a new codebase.

? Where would you like to create your turborepo? turbo-demo
? Which package manager do you want to use? yarn

Downloading files. This might take a moment.

>>> Created a new Turborepo with the following:

apps
 - apps/docs
 - apps/web
packages
 - packages/eslint-config-custom
 - packages/tsconfig
 - packages/ui

Installing packages. This might take a couple of minutes.

>>> Success! Created a new Turborepo at "turbo-demo".
Inside that directory, you can run several commands:

  yarn run build
     Build all apps and packages

  yarn run dev
     Develop all apps and packages

  yarn run lint
     Lint all apps and packages

Turborepo will cache locally by default. For an additional
speed boost, enable Remote Caching with Vercel by
entering the following command:

  yarn dlx turbo login

We suggest that you begin by typing:

  cd turbo-demo
  yarn dlx turbo login

baltez@Johns-MacBook-Pro ~/Code/demo
```

Any Questions?

Contact me

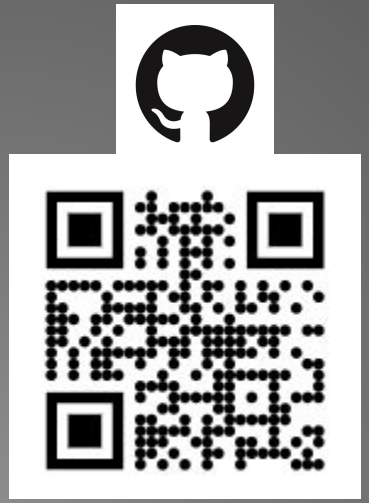
GitHub: retrojb

X: @johnbaltes6

LinkedIn: john-baltes

Email: baltescartography@gmail.com

Huge shout out to Reddit for the memes in r/programming Turbo Repo, Lerna, QR Code Generator, monorepo.tools,

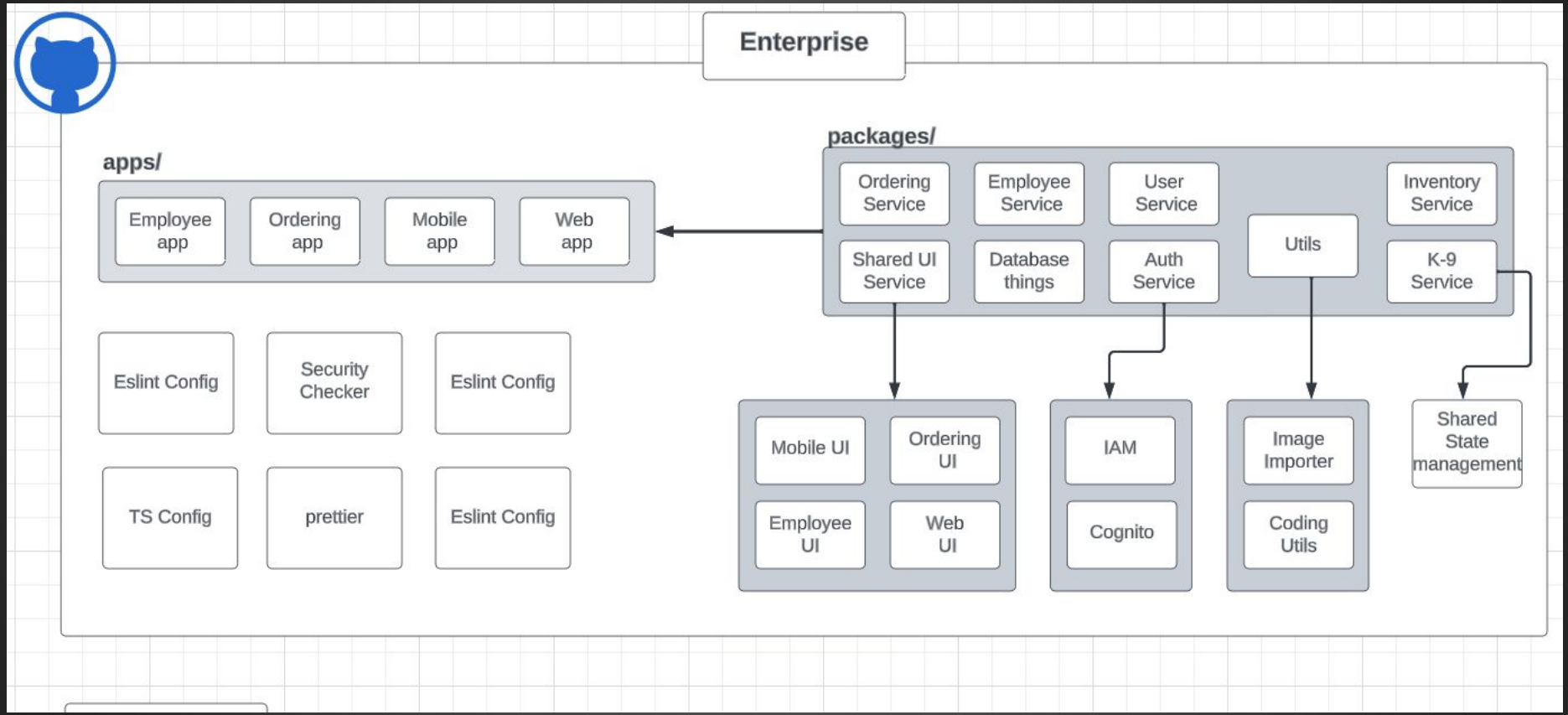


https://monorepo.tools/#what-is-a-monorepo

https://medium.com/tech-bits-pub/why-you-should-consider-monorepo-and-nx-for-your-project-f4993e95de3f

https://lerna.js.org/docs/introduction

https://turbo.build/repo/docs/installing


Enterprise
apps/
Employee app
Ordering app
Mobile app
Web app
packages/
Ordering Service
Employee Service
User Service
Inventory Service
Shared UI Service
Database things
Auth Service
Utils
K-9 Service
Eslint Config
Security Checker
Eslint Config
TS Config
prettier
Eslint Config
Mobile UI
Ordering UI
Employee UI
Web UI
IAM
Cognito
Image Importer
Coding Utils
Shared State management