The Ubiquitous nature of Javascript

Stephen White - 20200220
Göteborg React JS Meetup

CALLISTA

— ENTERPRISE —

Recorded Future

Meetup

ReactJS Göteborg

Göteborg, Sweden

1,025 members · Public group

Organized by **Stephen white** and **1 other**

Agenda

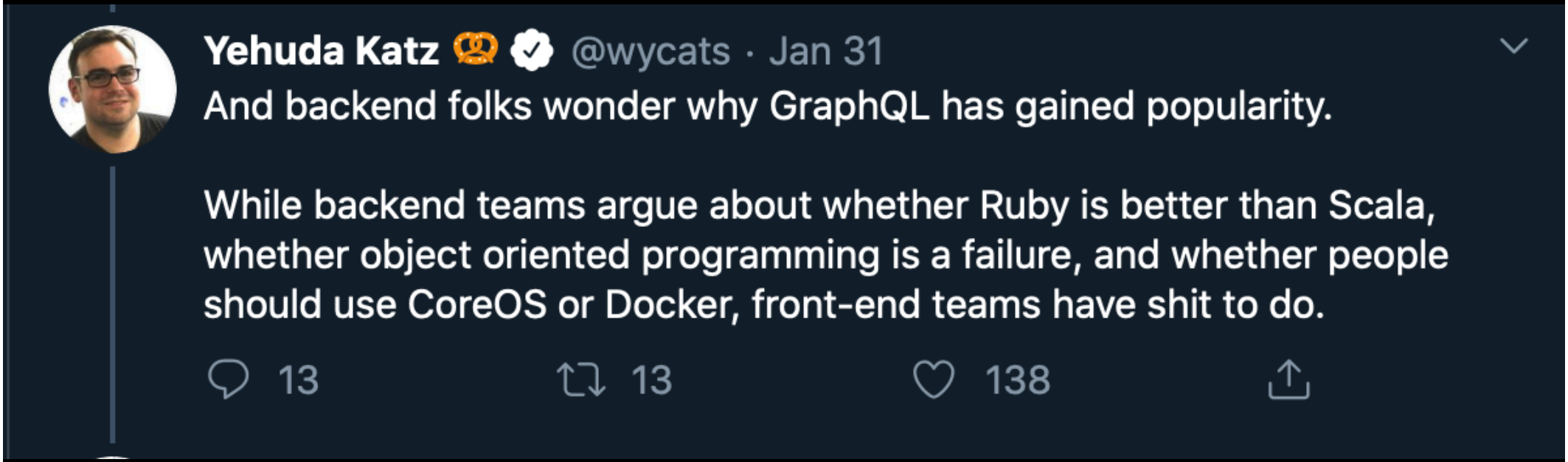
- Inspiration
- Dream Stack
- DX
- Backend for the Frontend - Serverless Framework
- Polly Dictate - landscape and code
- Demo

Inspiration

- The 60Fps syndrome- Yehuda Katz
- AWS Certification
- Rick and Morty

The 60fps syndrome

The 60fps syndrome



The 60fps syndrome

Different Concerns

- BE - Non Functional Requirements
- FE - Functional Requirements

The 60fps syndrome

Backend

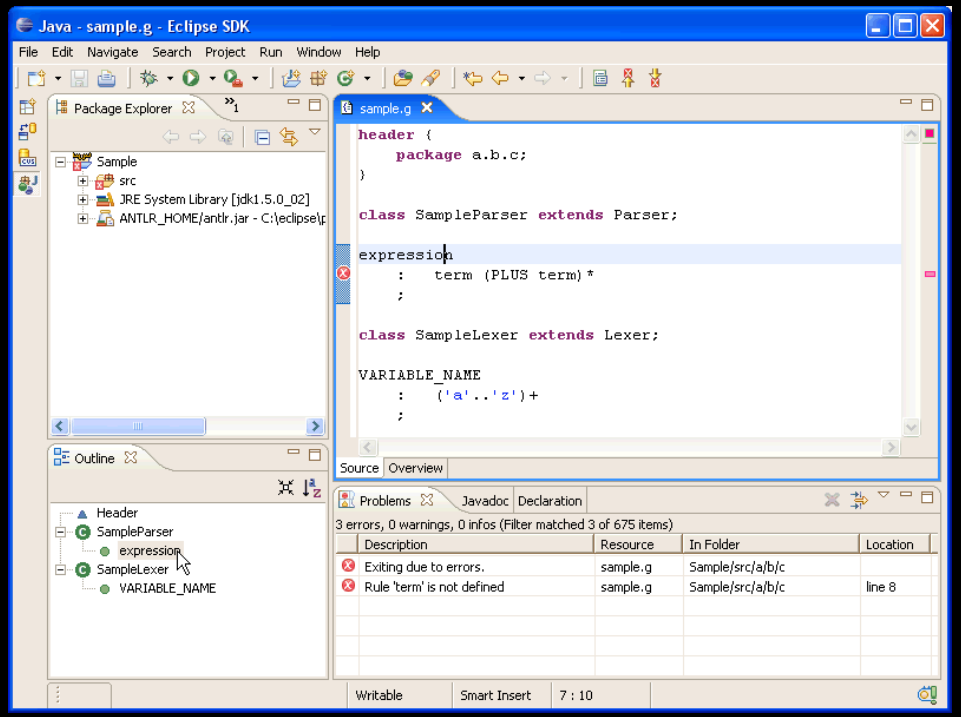
- BE - Non Functional Requirements
  - Accessability
  - Availability
  - Durability
  - Performance
  - Resilience
  - Scalability

The 60fps syndrome

Frontend

- FE - Non and Functional Requirements
  - Accessibility
  - Availability
  - Durability
  - Performance
  - Resilience
  - Scalability
  - Get shit done…

AWS Developer Cert



[as]

Two Brothers in a Van

Hindrance to 60fps

https://www.youtube.com/watch?v=Irc_dew1eYw

The Backend!

API’s - Rest vs GQL

BE Strikes back

Context Switching

The Process!

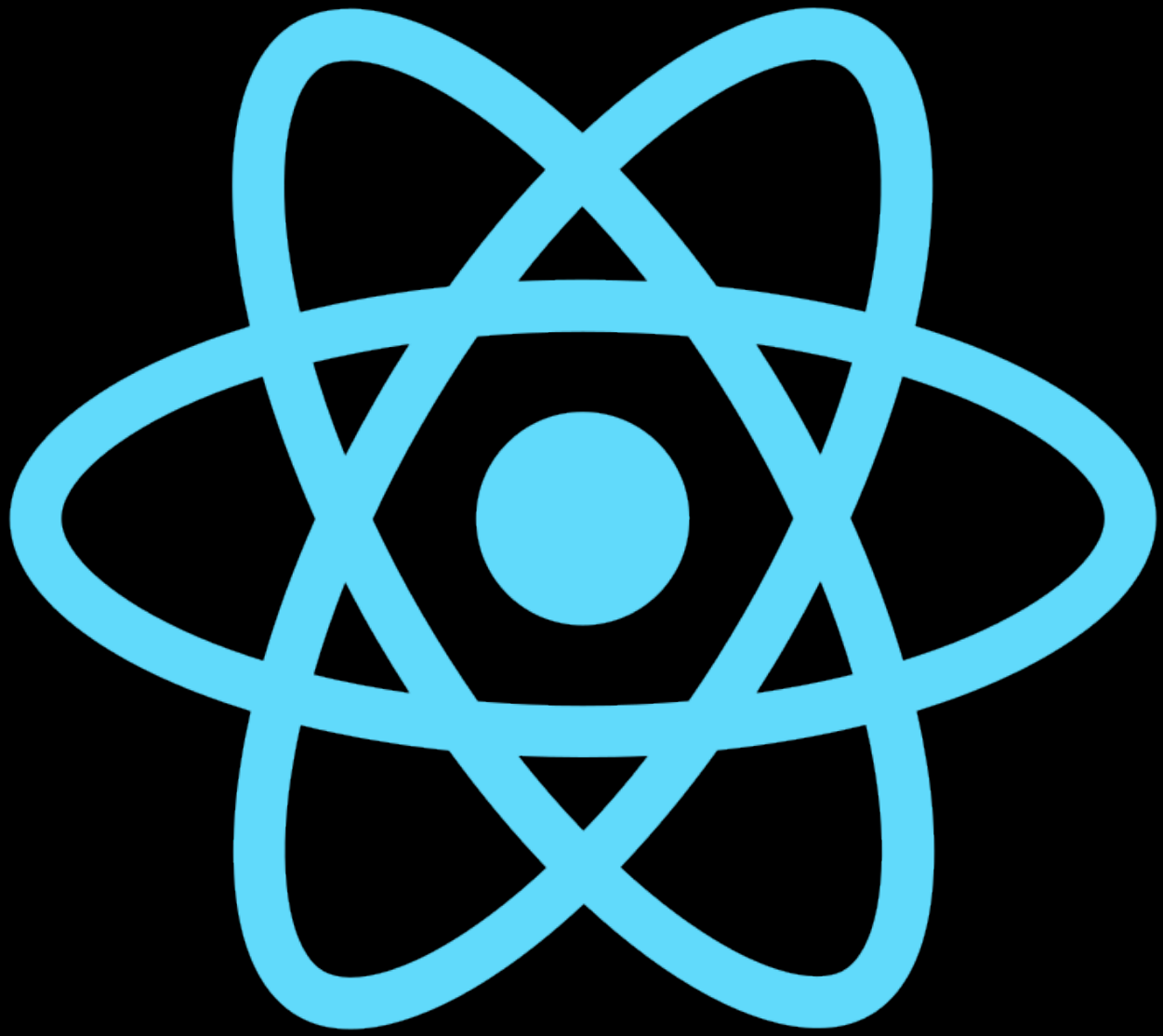
Technical Dept

If only we could use JS in the BE

Dream Stack

serverless

framework

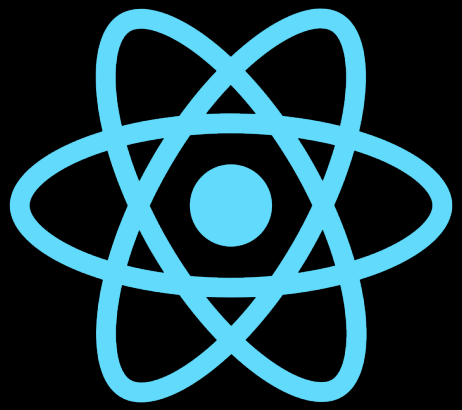


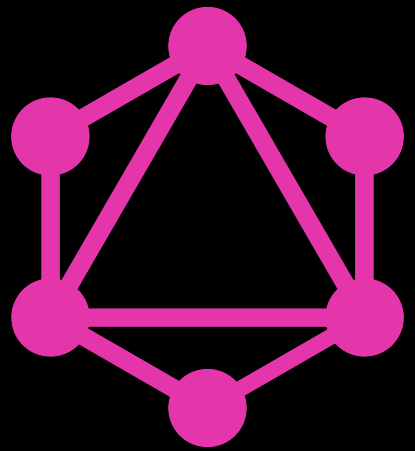

DX


DX


DX

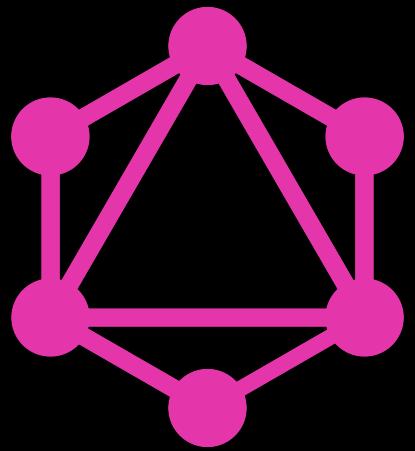
Frontend Patterns



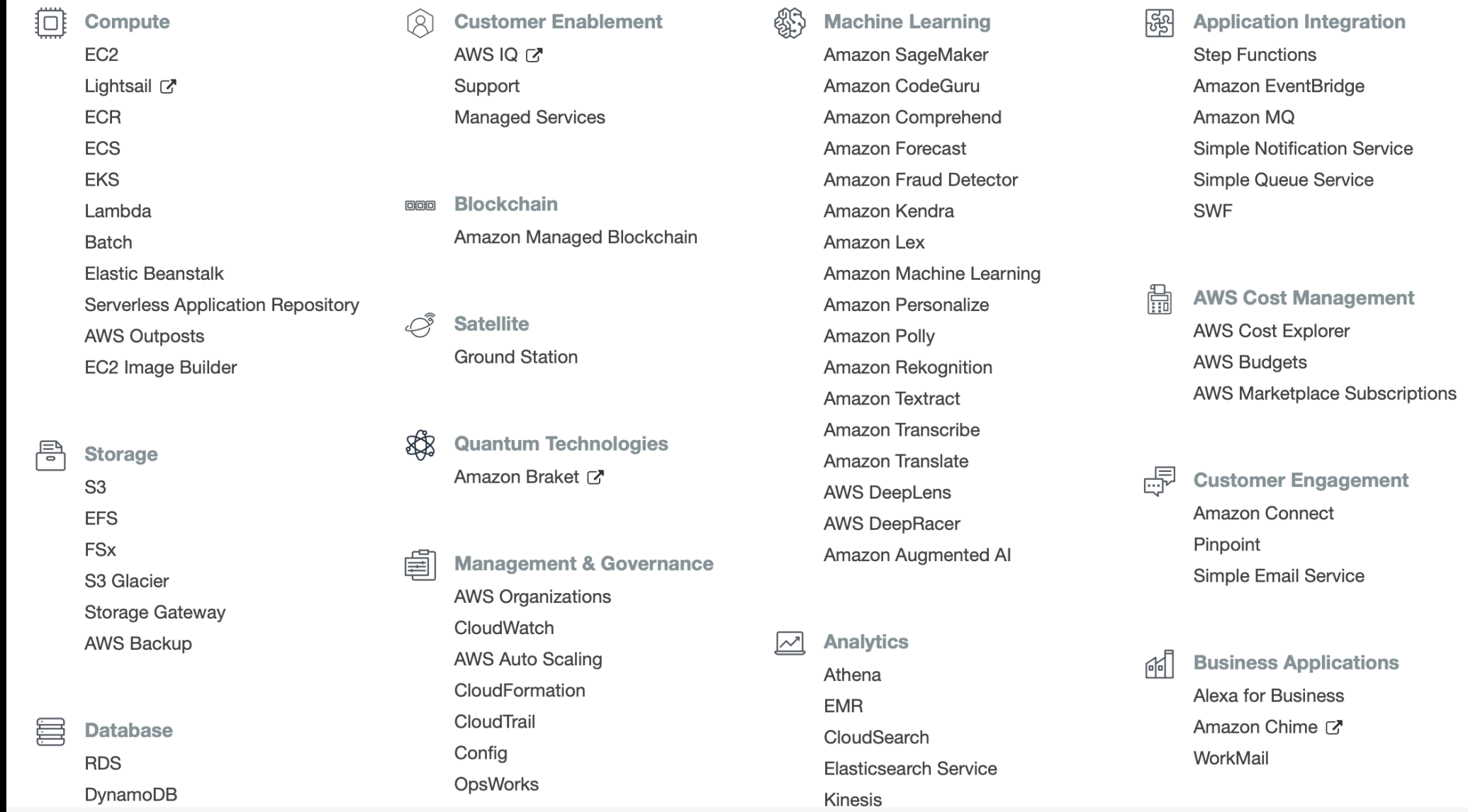


Backend Patterns

Almost the same



Although where did all those AWS Services come from!!



Serverless and JS gives us

- DX
- Minimal Context switch
- Tooling
- Code as Infrastructure
- Patterns and Conventions
  - Functional Programming

Move towards Cross Functional Teams

Serverless Framework, what it offers

- JS is king!
  - JVM based languages, Kotlin, Java … Scala
  - GO
  - Rust
- Code as Infrastructure
- Plugin Model
  - Offline
  - Webpack - hot reload
  - SNS / SQS - Offline!!
  - ….
- Deploy to multiple clouds (Although most support for AWS)

Lambda

```
export const get = async event => await ns.get(event.path.id)
```

Thin as possible

Service

Most of your code

Notes DB

Code As Infrastructure

```yaml
notes-get:
  handler: src/notes.get
  environment:
    TABLE: ${self:custom.pollynotesDb}
  events:
    - http:
        path: notes/{id}
        method: GET
        integration: lambda
        cors: true
        request:
          parameters:
            paths:
              id: true
        authorizer: aws_iam
```

API GW

/notes/{id}
GET

GetNote

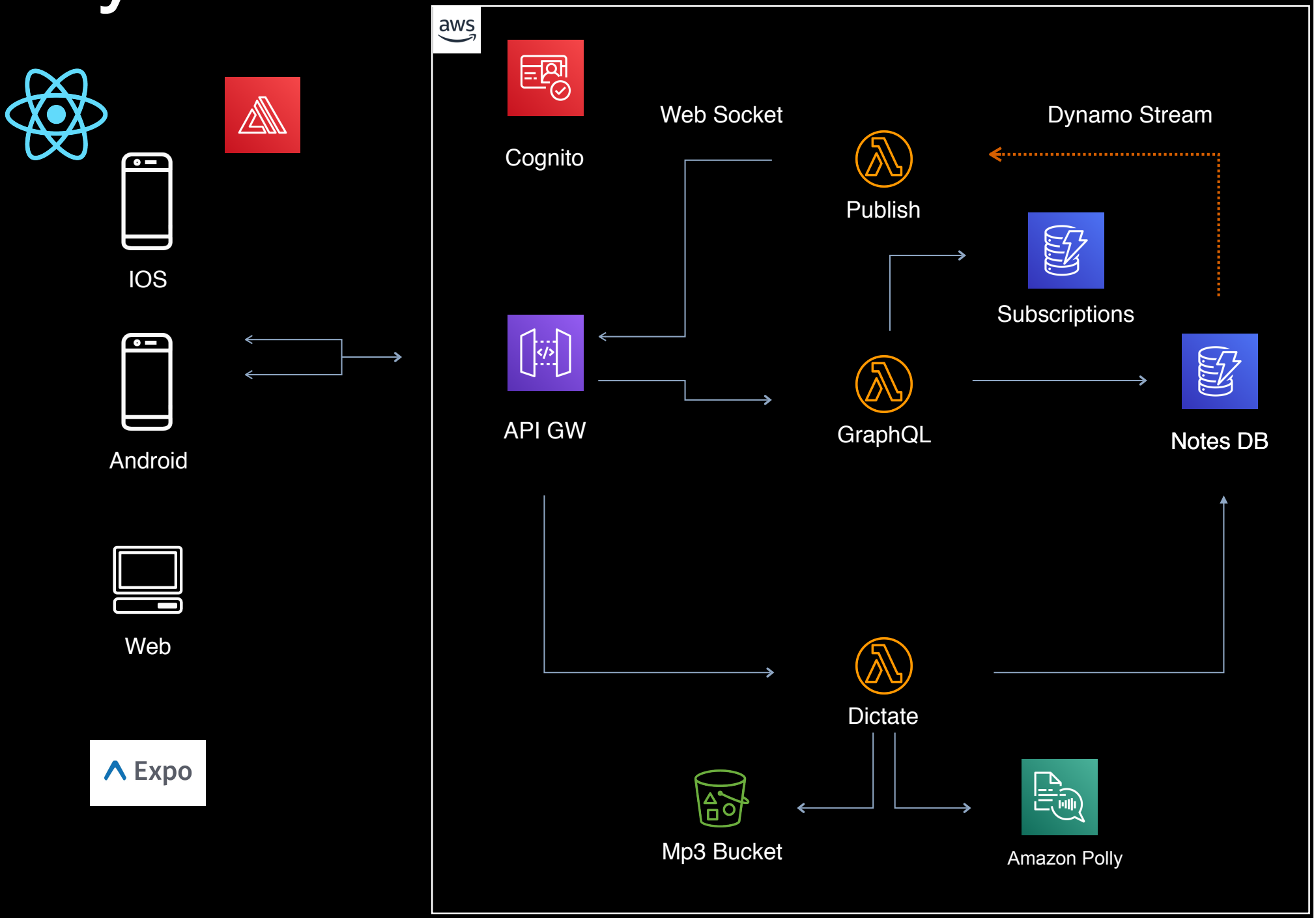
Test

```
it("can create a note", async () => {
  const note = await api.post({ note: "the cat on the mat" });
  expect(note.note).toEqual("the cat on the mat");
  await api.delete(note.id);
});
```

Local Remote

Demo

Polly Dictate

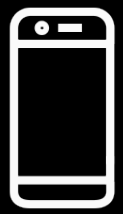


Expo - deploy

```
App.json
"expo": {
  "bundleIdentifier": “polly-native-expo”,
  "updates": {
    “enabled”: true
    "checkAutomatically": true
  }
}
```

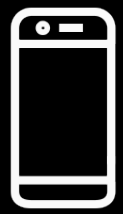
Expo App

```
App.json
"expo": {
  "bundleIdentifier": “polly-native-dev”,
  "updates": {
    “enabled”: true
    "checkAutomatically": true
  }
}
```



Dev App

```
App.json
"expo": {
  "bundleIdentifier": “polly-native”,
  "updates": {
    “enabled”: false
    "checkAutomatically": false
  }
}
```



Prod App

Channels	Default	Develop	Production

expo publish --release-channel <channelName>

Conclusions

- Two Bros
  - The backend, move away from functional teams to cross functional with a common language… JS
  - API's Rest vs GQL, let the api be driven by the needs of the client not by what the BE have time to provide. Make it flexible and let it evolve.
  - BE Strikes back! Talk the same language have the same concerns (i.e the customer)
  - Context Switching - mitigated with the use of JS in all layers and the use of a decided programming style, be it functional or OOP.
  - The Process - with greater efficiency and productive, trust and empowerment of the developer follows, less trust generally means more process, more trust infers that the devs can focus on getting stuff done!
  - Technical Dept - will decrease if more devs take responsibility for the code base. With greater DX devs have more time to fine tune and polish their code.
  - JS in all levels!

Conclusions

- Two Bros - The Process
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan
  - https://agilemanifesto.org

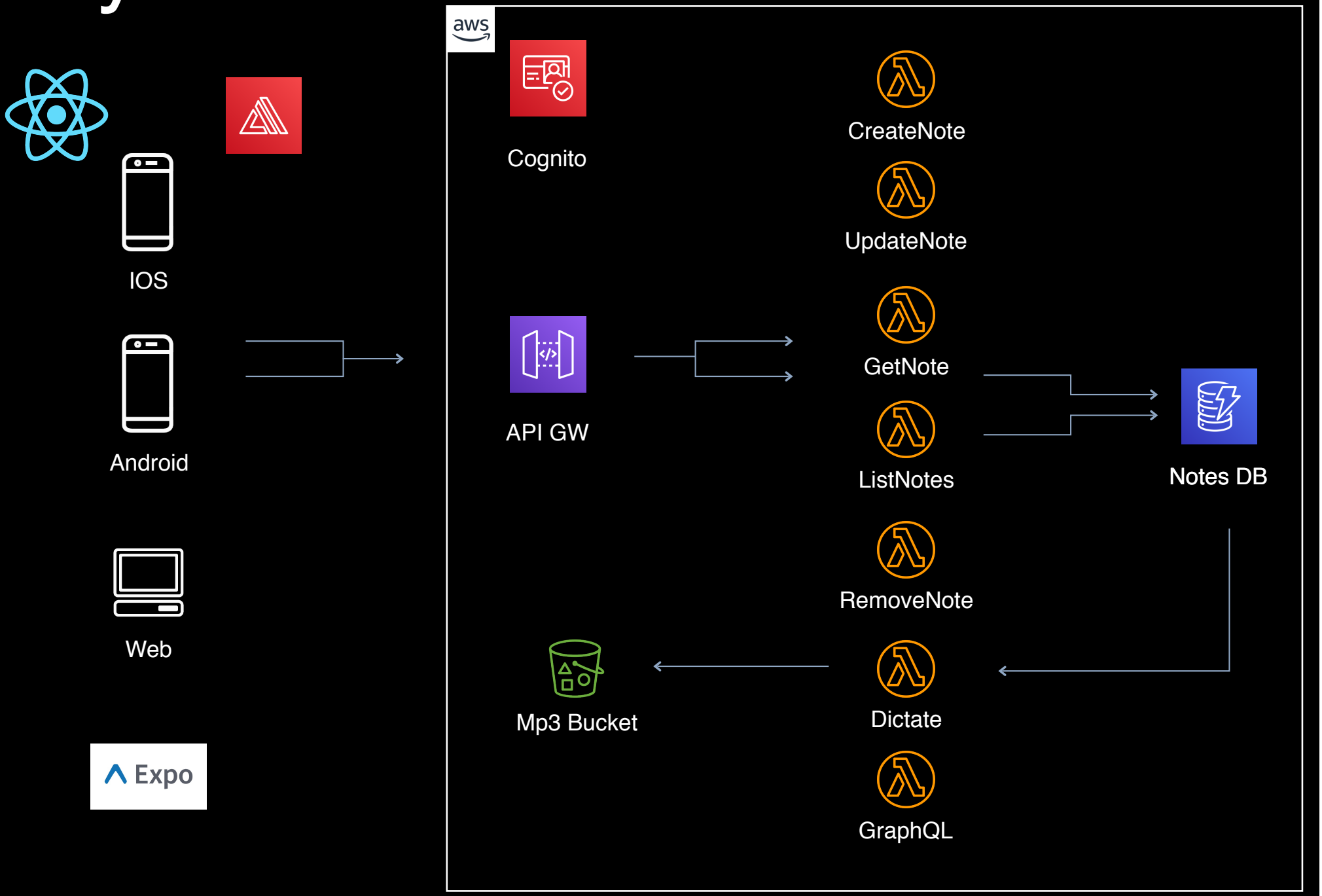
Conclusions

- React Native
  - gives a native experience.
  - Web skills for native apps.
  - Great community
  - Modern standards

Demo List

- Show the app in IOS and Android Sims
- Show the Web App
- Run the BE locally
  - Show GraphIQL
  - Add a gql resolver
    - SearchNotes
  - Back to GraphiQL

Polly Dictate



Yehuda Katz 🥨 ✓

@wycats

This back and forth about microservices vs monoliths is hopelessly stuck in a backend-focused world.

What else should we expect? Front-end engineers are given the incredibly hard tasks of caring for the end user. In too many places, backend engineers still control everything.

2:14 AM · Jan 31, 2020 · Twitter for Android

Yehuda Katz 🥨 @wycats · Jan 31

And front-end engineers have to fight for crumbs of support from the hallowed halls of the backend team.

I once watched a front-end team spend years asking their backend team for a single field to store arbitrary data in. Sorry, the big brains in the backend team know best.

11 4 78


Yehuda Katz @wycats · Jan 31
We're far overdue for a conversation about web application architecture where front-end developers took a seat at the table, and where our concerns mattered as much as the latest iteration of the argument about monoliths vs microservices.
15
15
123

Enterprise Damage