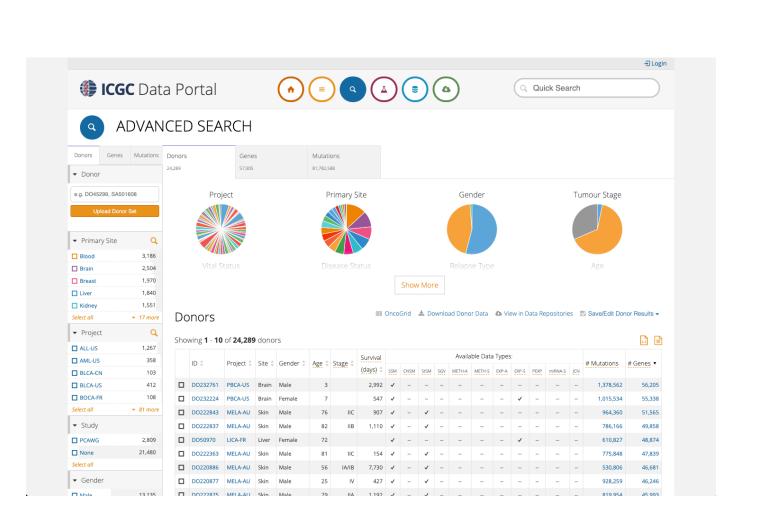
# Conquering Micro-services with GraphQL

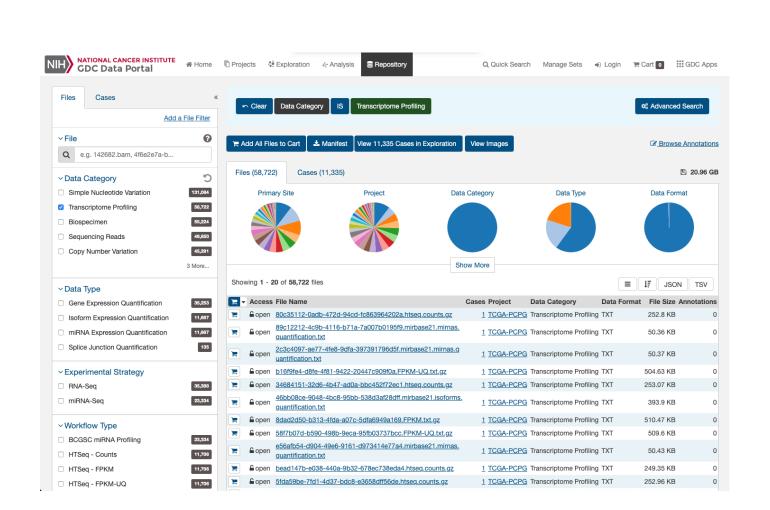
Our plan for Argo



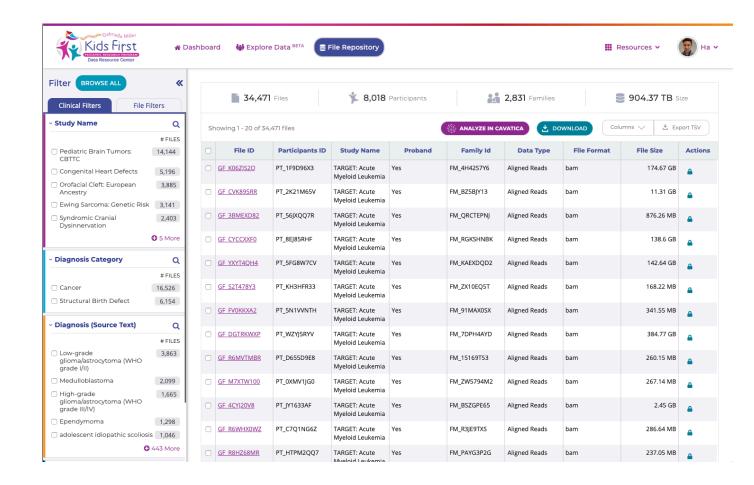






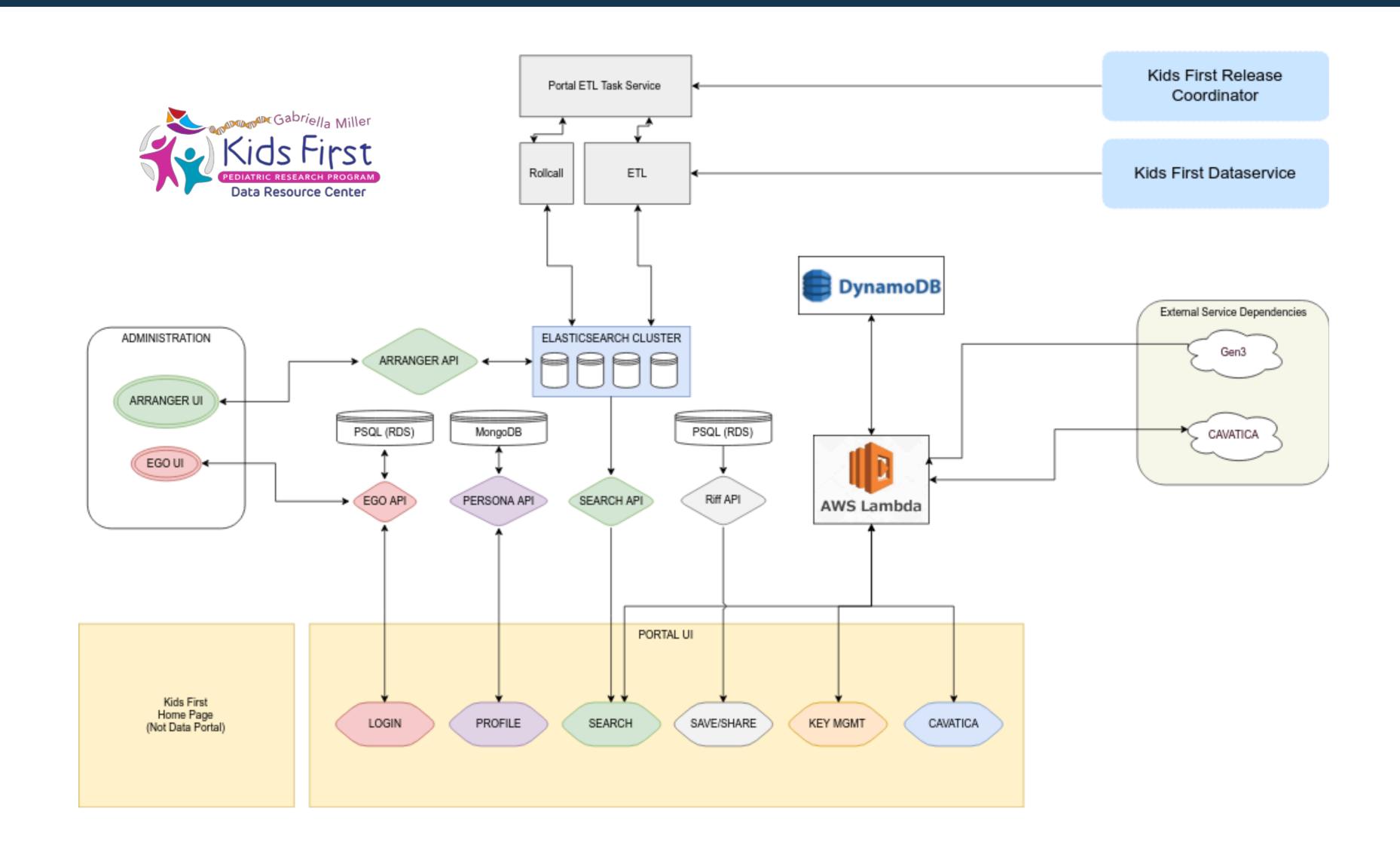




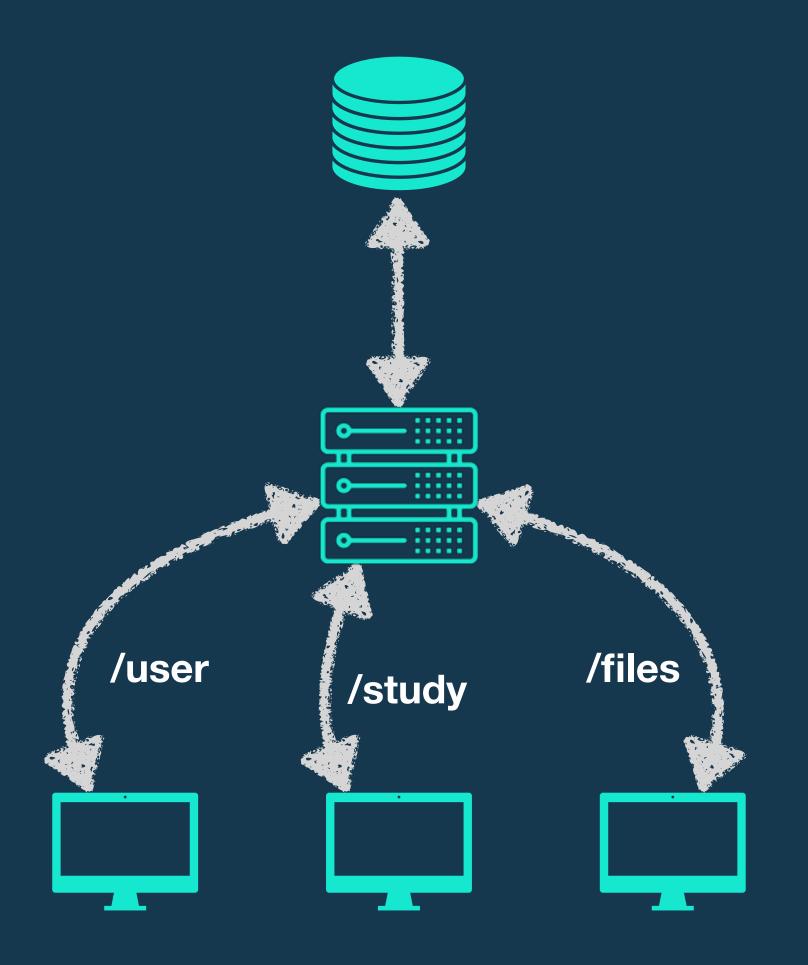


and more...

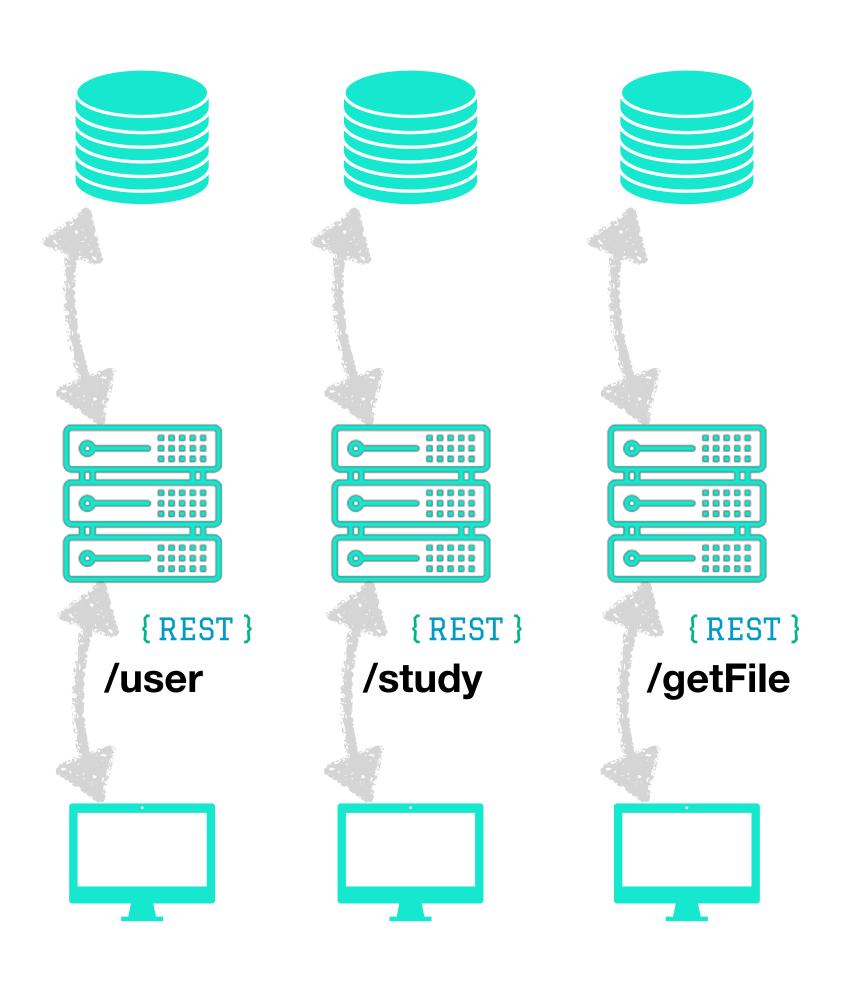
#### Behind the scene



## Monolithic System



#### Micro-...what?

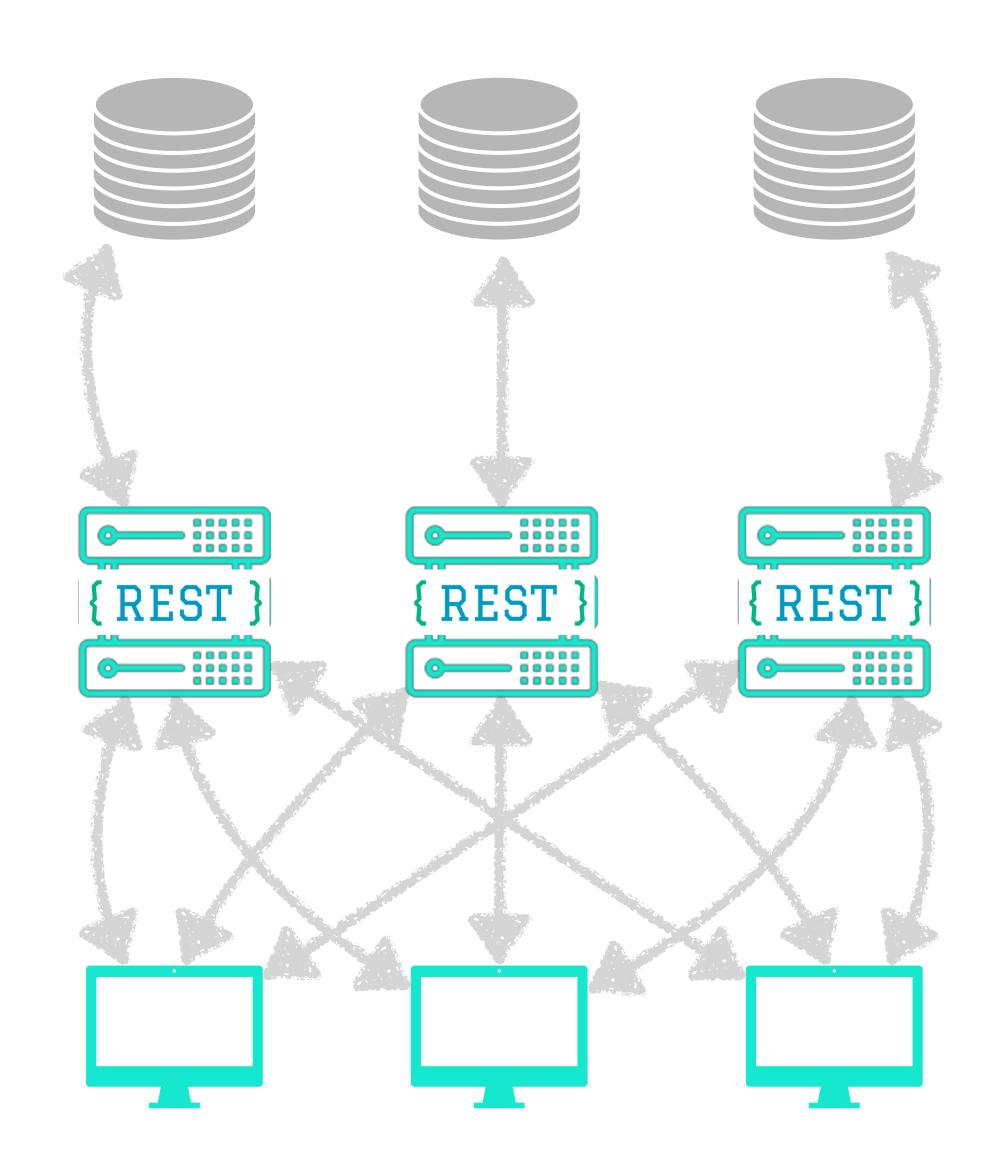


# Micro-services System

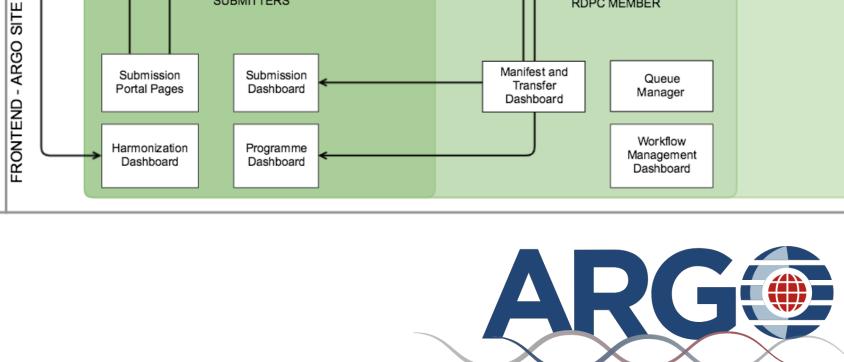
- Functionally decoupled
- Independently deployable
- Independently scalable

### Some challenge

- No place to handle cross-cutting concerns
- Front-end handles heavy business logic
- Inefficiencies with REST:
  - Over & under-fetching
  - Lack of API schema -> manual documentation







SONG

SCORE

Big Data Store

(clinical / workflow

file results,

Authorization, Authentication and Identity

Management (EGO)

Data Access

Control

(DACO)

RDPC MEMBER

annotation files)

Program Database

Submission System

Validation

System

Data

Dictionary

Clinical SONG

RDPCs

RDPC

Datastore

SONG-R1

SCORE-R1

Workflow

System

MIDDLE WARE

Centralized Services

ID Service

GENOMIC DATA -

SUBMITTERS

R2

RDPC Datastore

SONG-R2

SCORE-R2

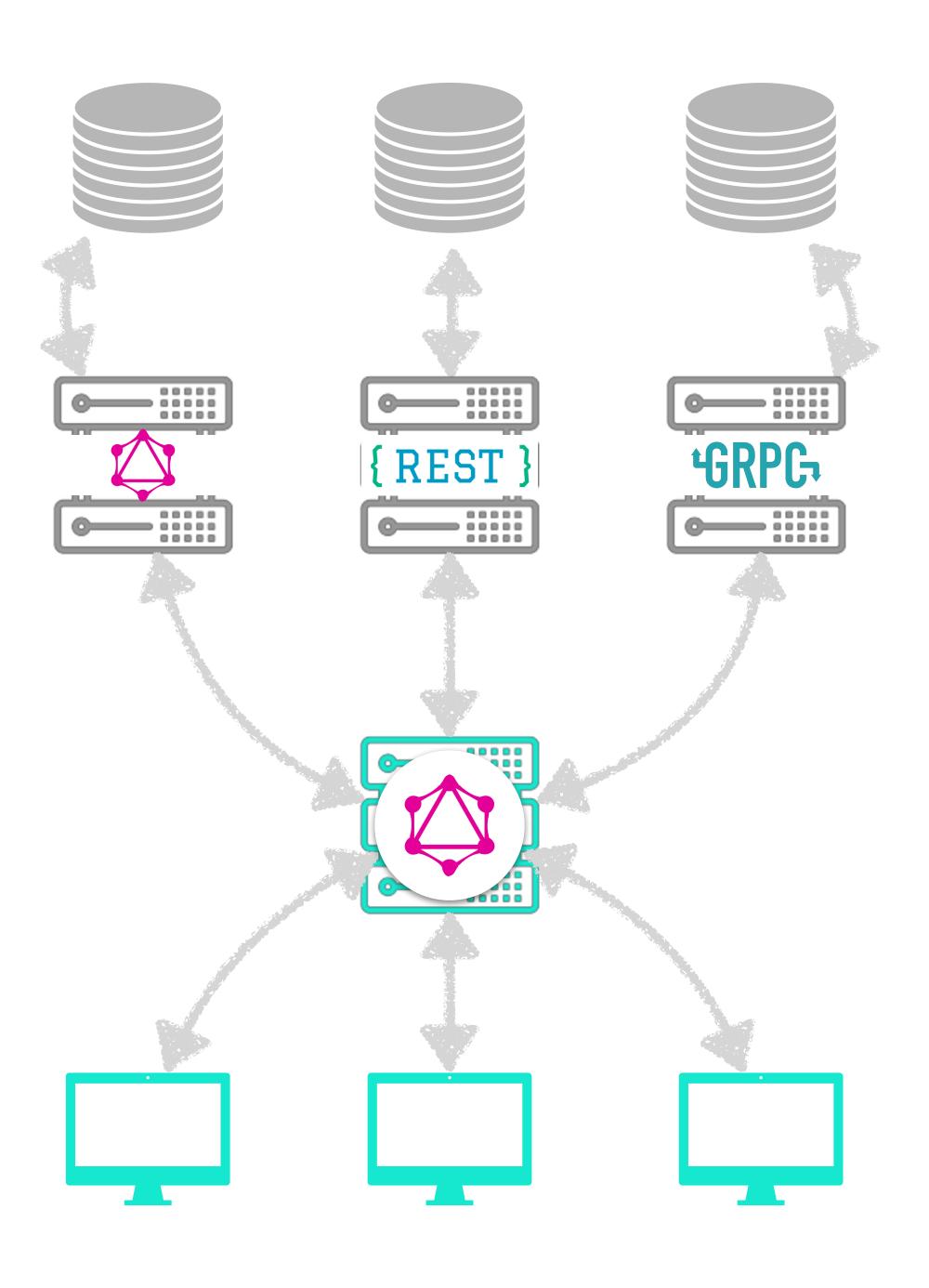
Workflow

System

Location

Policy Engine

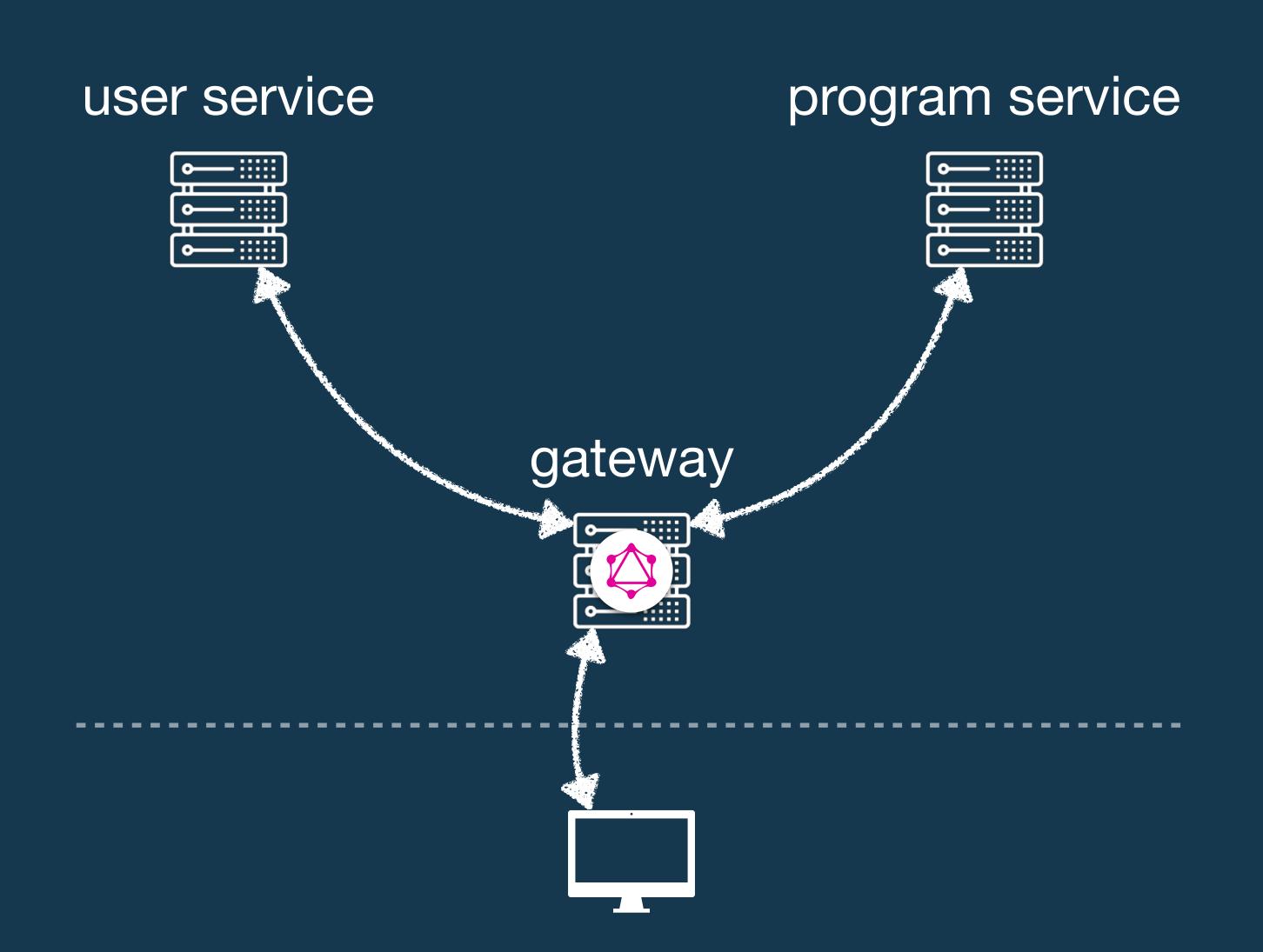
CLINCAL DATA -



### The GraphQL Solution

- •An aggregation layer to handle cross-cutting concern
- ·Why GraphQL?
  - Explorable schema —> selfdocumenting
  - Schema Stitching:
    - •Modelling underlying services as an independent schemas
    - "Stitch" the schemas together as nodes in a graph
    - Network advantage: batched request

### demo



thank you