GRAPH QL

STEPHEN.WHITE

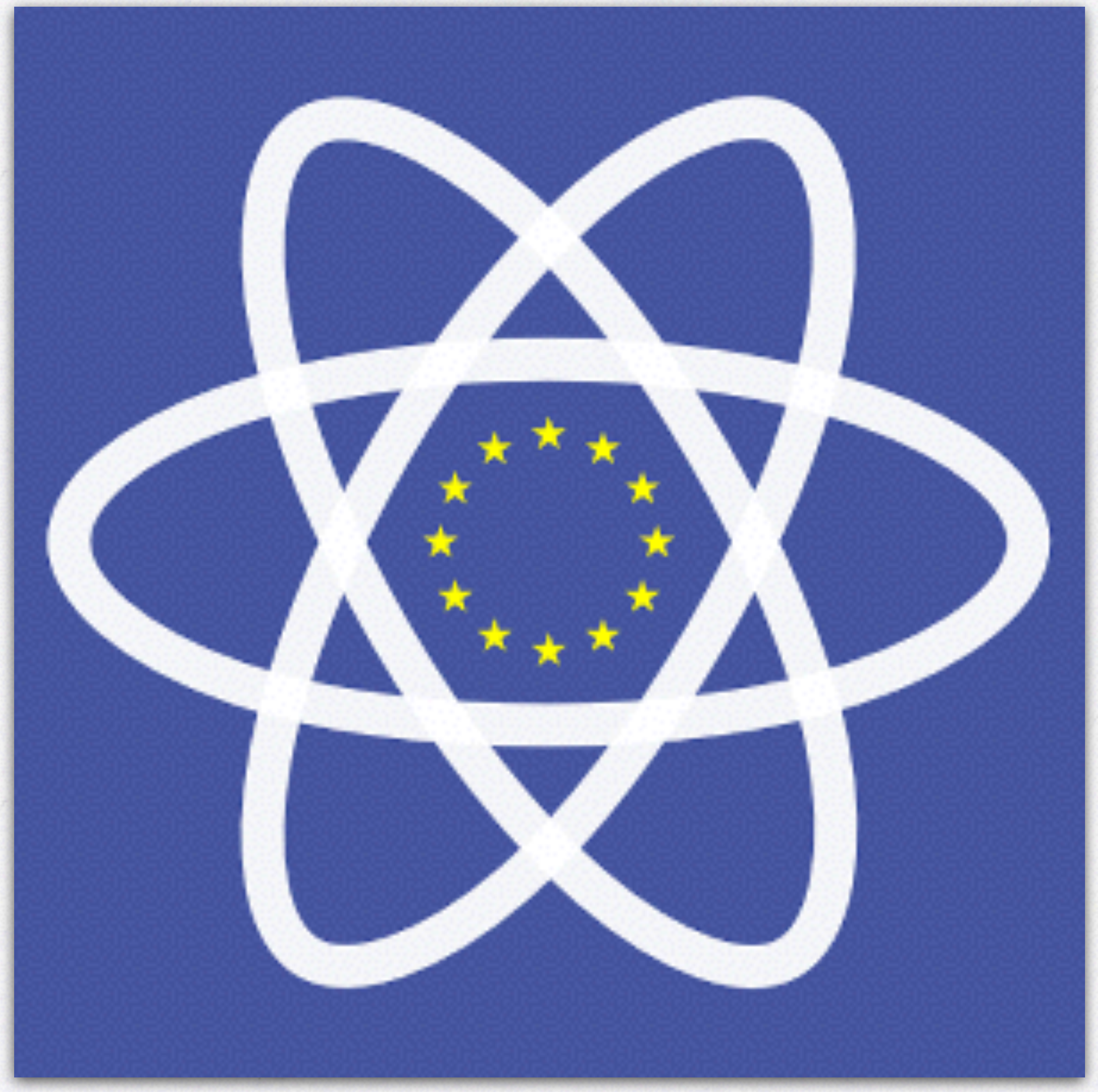
| CALLISTAENTERPRISE.SE

GRAPH-QL

AGENDA

- My Interest
- What is Graph QL
- Demo
- Why GraphQL
- Summary

REACT EU CONF



POINTS TO REMEMBER

DX

Developer Experience

POINTS TO REMEMBER

Reason

about your code

MEETUP QUOTE

React just vanishes into your code

MEETUPS - GÖTEBORG REACTJS

ReactJS Göteborg

http://www.meetup.com/ReactJS-Goteborg/

MEETUPS - LONDON REACTJS

http://www.meetup.com/London-React-User-Group/

AND SERIOUSLY!

260 BILLION!

Requests a day!

What is it?

WHAT

It's a **graph query language**

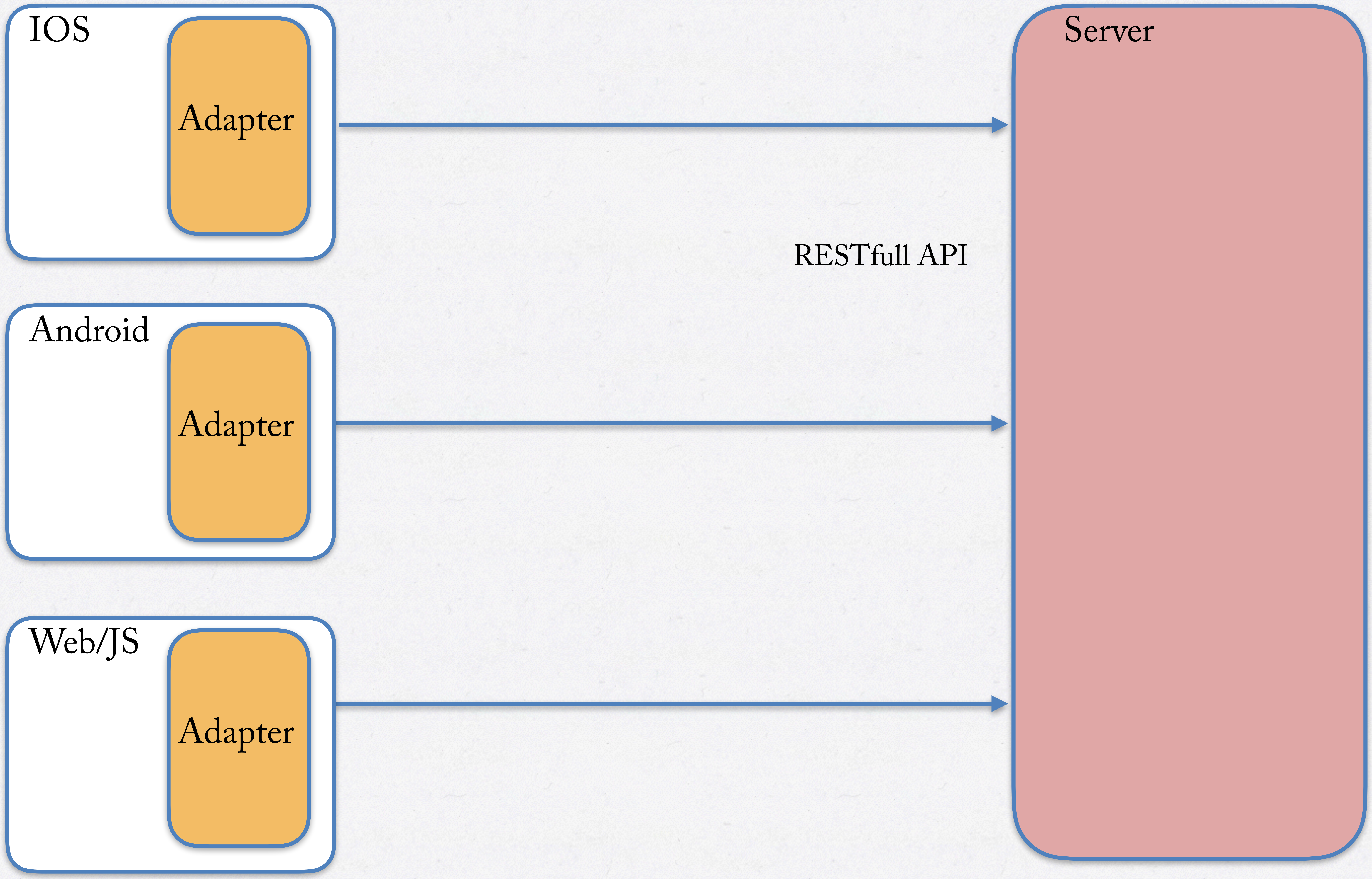
that presents the

Possibilities

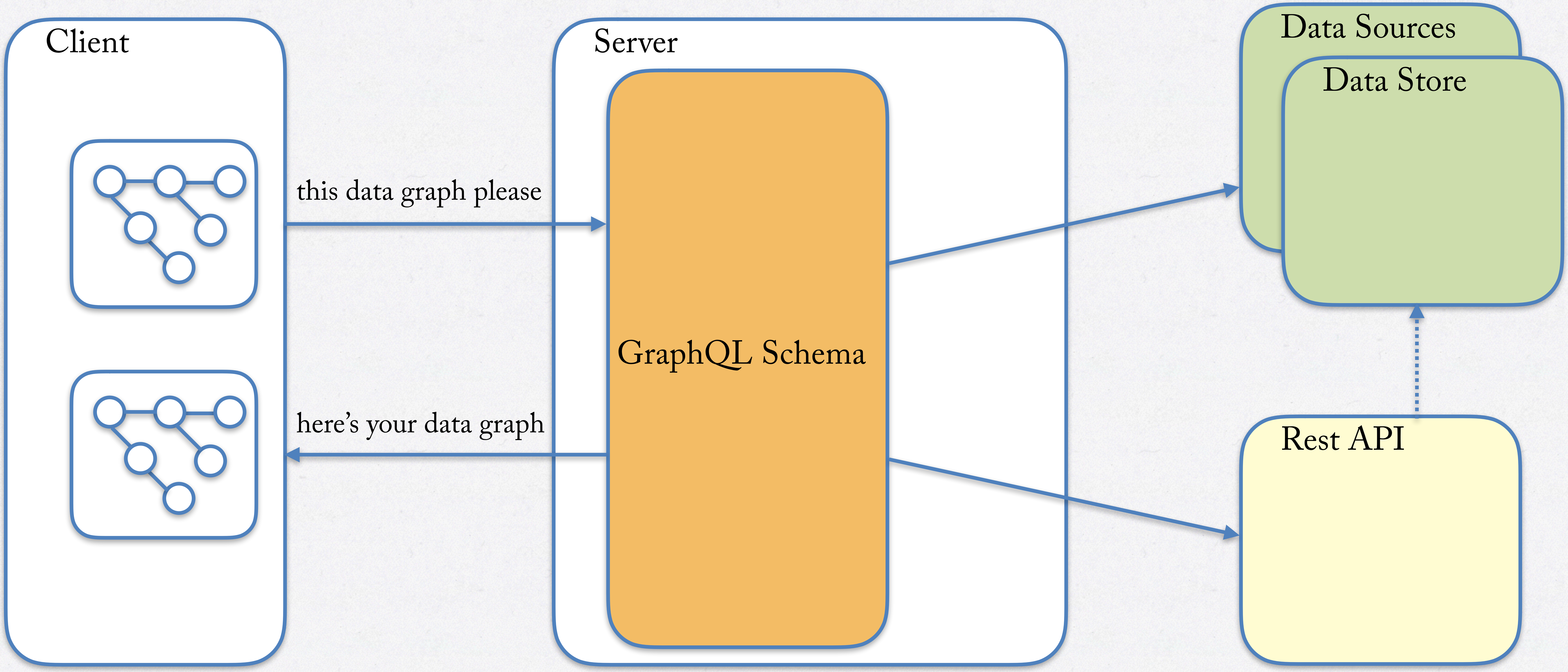
of your API

It's **not** a **Data Store** or **SQL**
But acts as
A Server side Translation
between your **client** *and your*
Data Store

TOO MANY ADAPTERS!



ONE ADAPTER TO RULE THEM ALL!



Mediates/Adapts/Aggregates

The client decides the format of it's data via it's queries

WHAT - IT PRESENTS YOU WITH

- Hierarchy
- A Product-centric approach (client)
- Client-specified queries
- Backwards Compatibility
- Structured Code, in the form of Composition
- Application-Layer Protocol
- Types
- A Method of Introspective

CORE PRINCIPLES

- Compossible
- Mental Model
- Graph Data
- Types
- Version Agnostic

Its Compossible

CORE PRINCIPLES

Posts

```
query{
  posts{
    _id,
    title
  }
}
```

returns :

```
posts :
[{ _id,title}]
```

Post

```
query{
 posts(_id:1){
    _id,
    title,
    body
  }
}
```

```
"posts": [
 {
   "_id": 1,
   "title": "ReactJS",
   "body": "redux, immutablejs"
 }
]
```

Comments

```
query{
  comments(postfk:1){
    _id,
    commentText: body
  }
}
```

```
"comments": [
  {
  "_id": 1,
  "commentText": “Redux?"
      },
  . . .
  ]
```

CALLISTA
— ENTERPRISE —

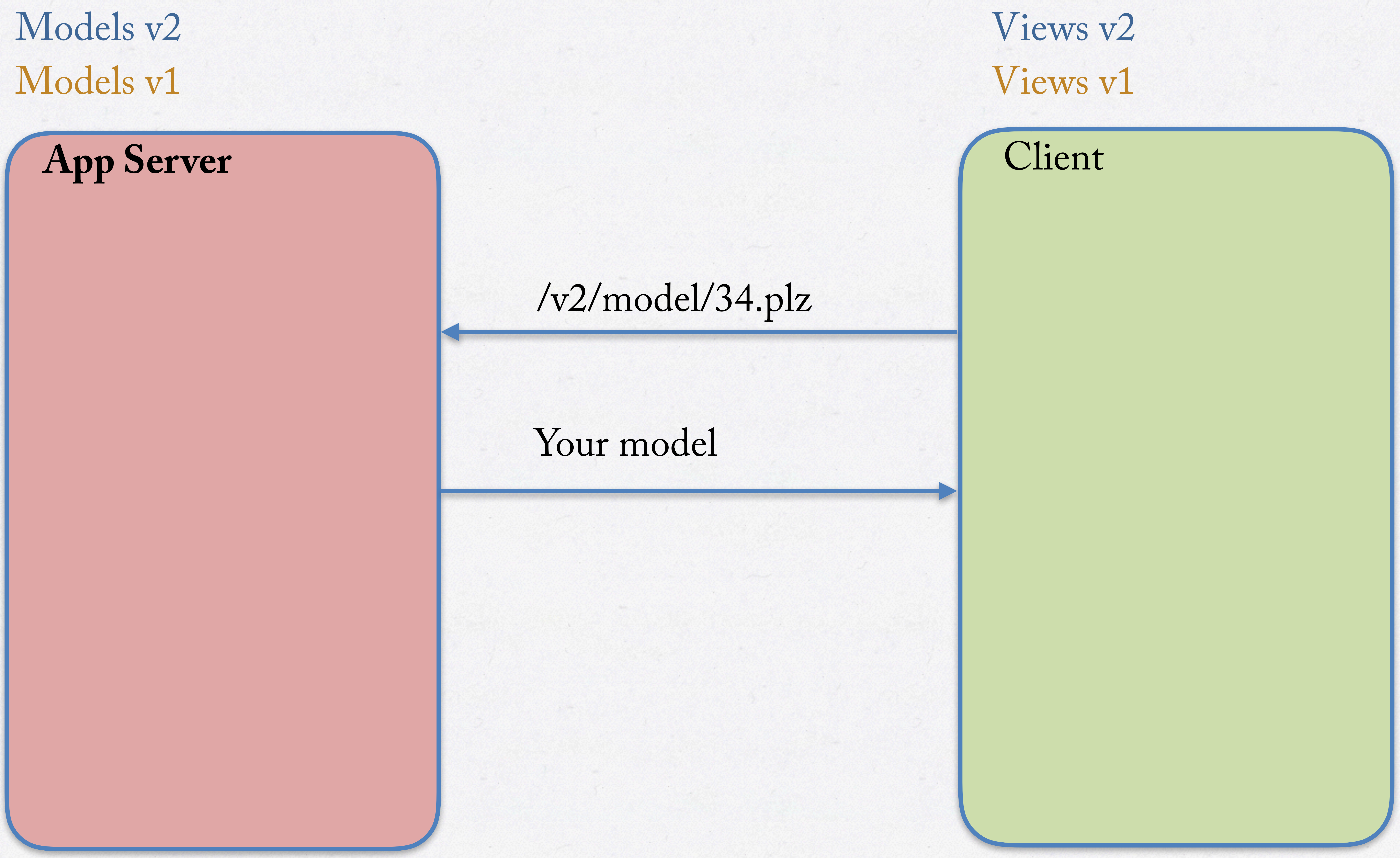
Types

Mental Model

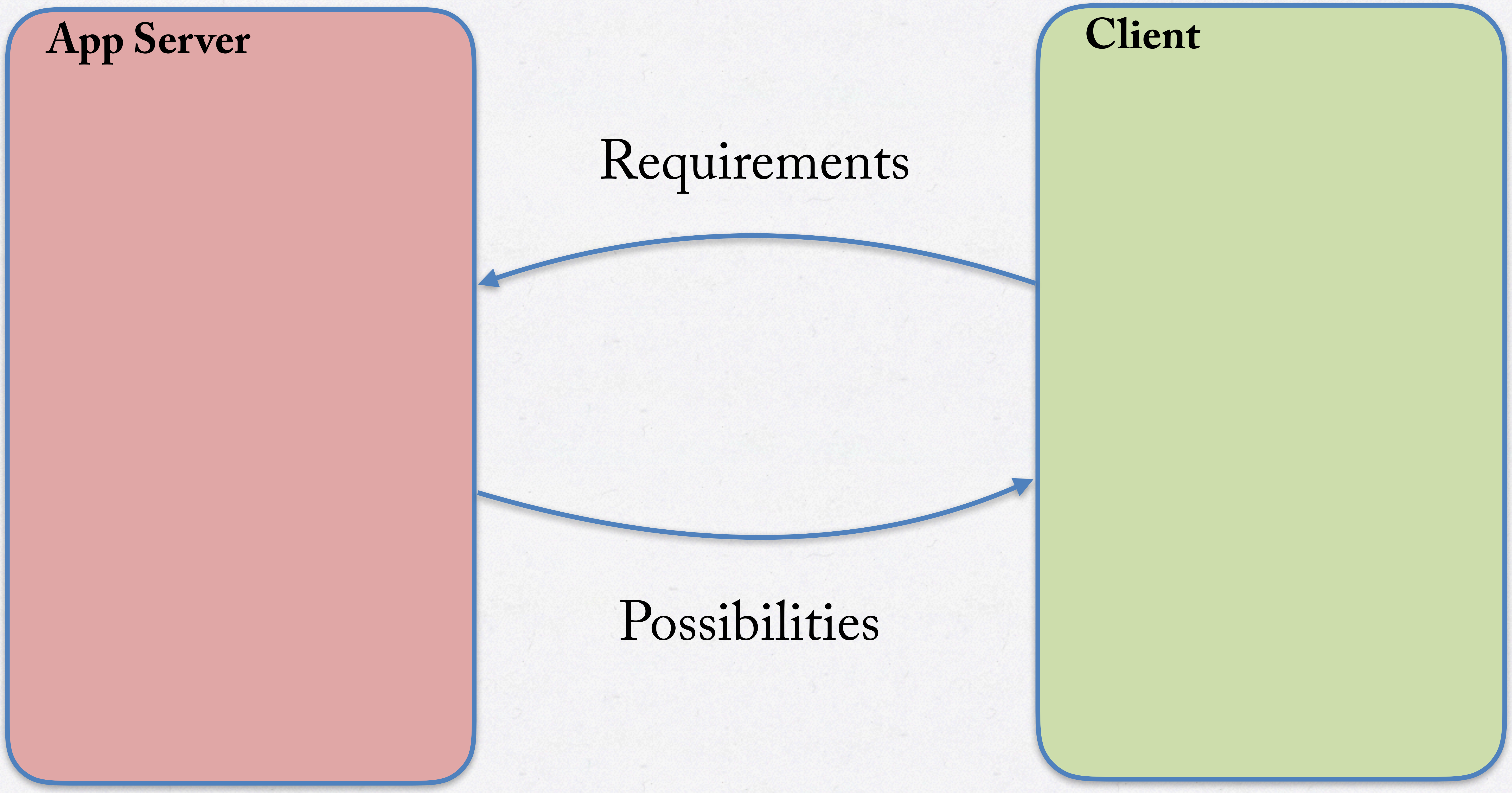
Graph of Data

Version Agnostic

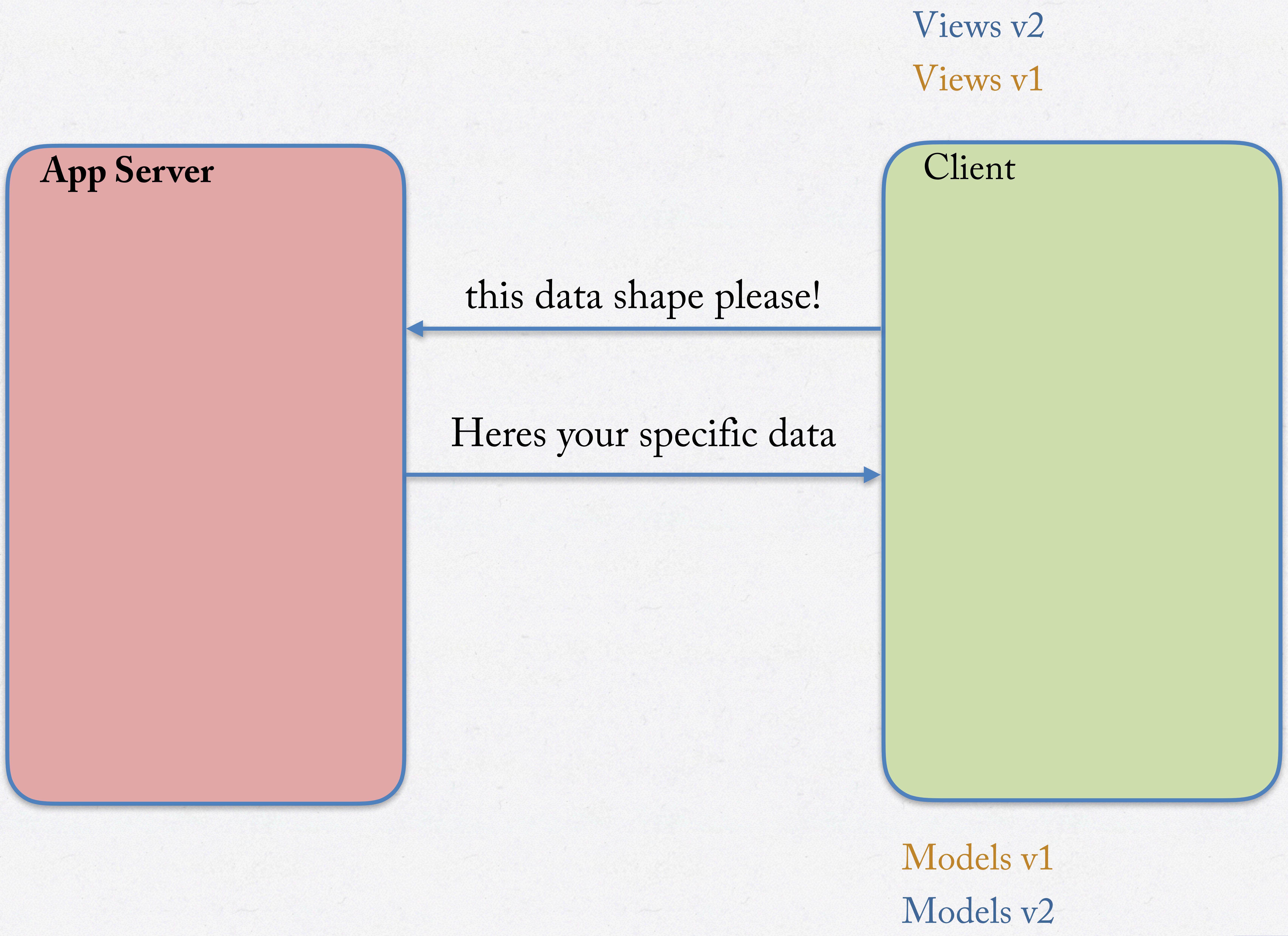
REST API


Models v2
Models v1
Views v2
Views v1
App Server
Client
/v2/model/34.plz
Your model

GRAPH-QL

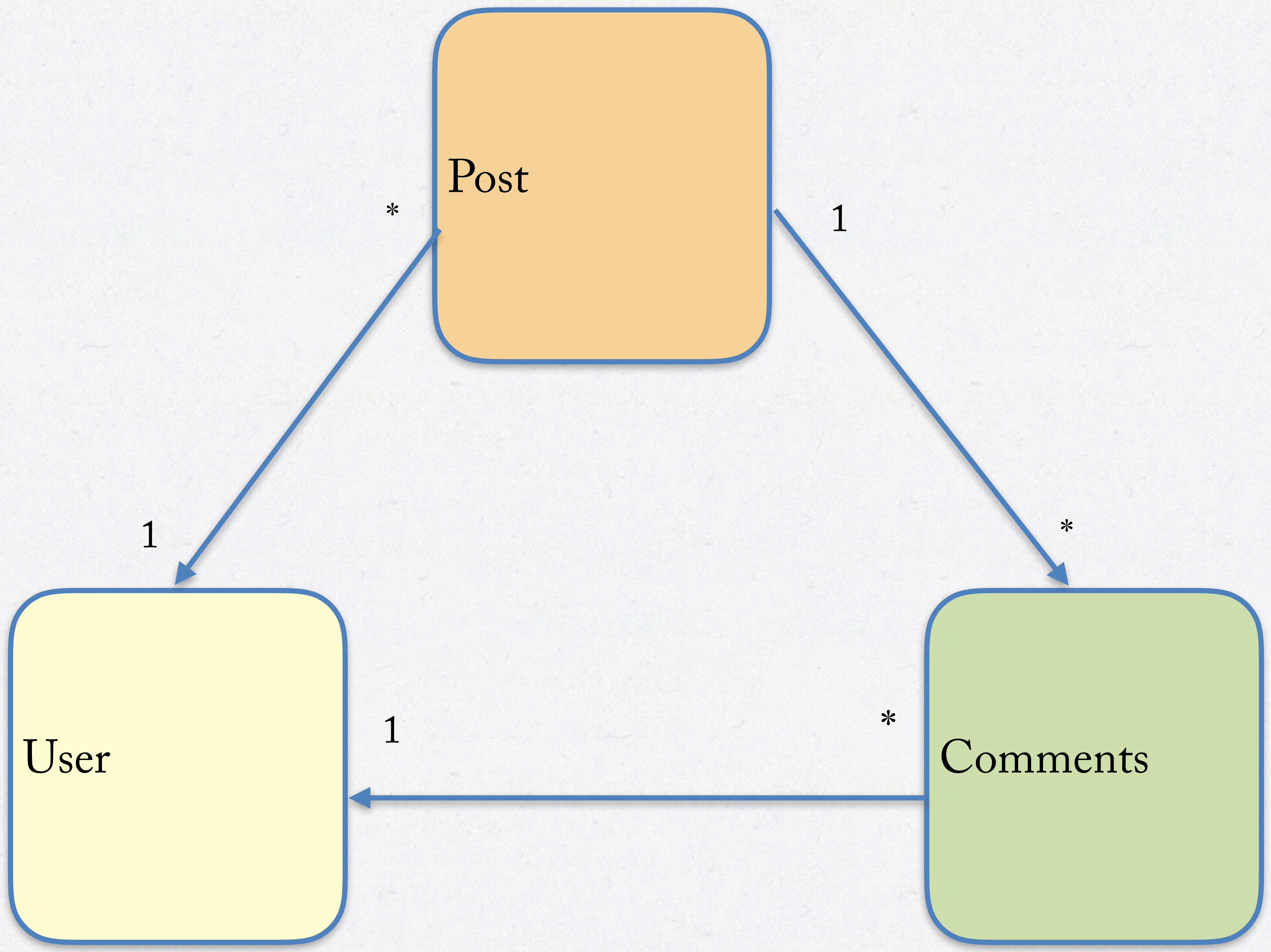


GRAPH-QL

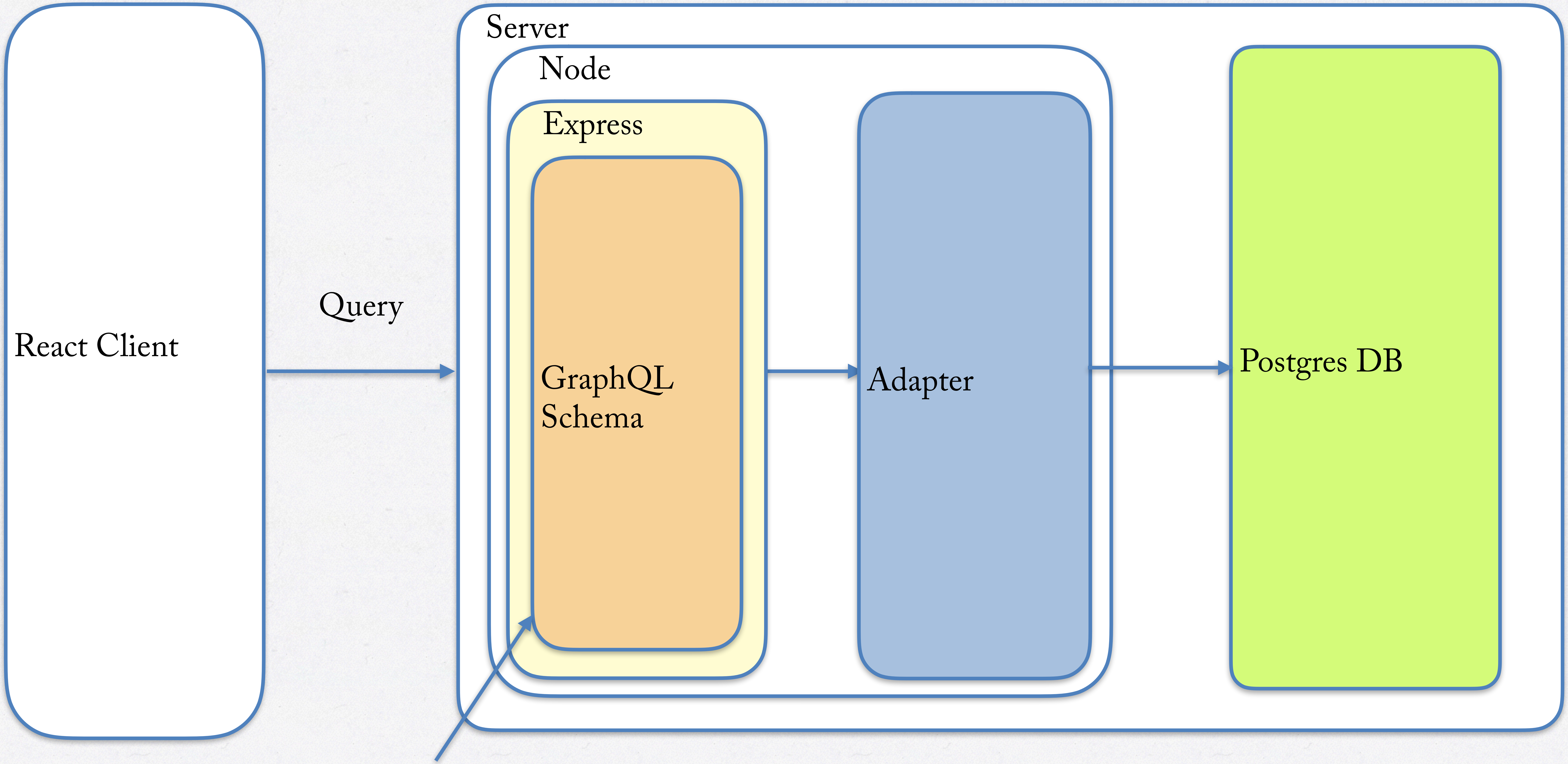

Views v2
Views v1
App Server
Client
this data shape please!
Heres your specific data
Models v1
Models v2

Technical Setup

MODEL



NODE JS IMPLEMENTATION



Defines the possibilities of your API maybe similar to an Interface

CLIENT CODE

React Client

```
// general query function, using fetch and returning a promise
export function query(query) {
  return fetch(gqlserver, {
    method: 'post',
    headers: { 'Content-Type':'application/graphql' },
    body: query,
  }).then((result, error)=>{
    return result.json();
  })
}
// defined query
export function getPost(id) {
  let postsQuery = `{posts(_id:${id}){_id,title,body}}`;
  return query(postsQuery);
}
// mutation to update a post
export function updatePost(post) {
  let jsPost = post.toJS();
  let addCommentQuery = `mutation { updatePost(_id:$
{jsPost._id},body:"${jsPost.body}"){_id,title,body,userfk}}`;
  return query(addCommentQuery);
}
```

CALLISTA
— ENTERPRISE —

GRAPHQL-SCHEMA

GraphQL Schema

```
// Types
const User = new GraphQLObjectType({})
const Post = new GraphQLObjectType({})
const Comment = new GraphQLObjectType({})

// Queries
const Query = new GraphQLObjectType({

// Mutations
const Mutation = new GraphQLObjectType({

// Schema
const Schema = new GraphQLSchema({
  description:’my scheam',
  query: Query,
  mutation: Mutation
});
```

CALLISTA
— ENTERPRISE —

GRAPHQL-ADAPTER

Adapter

```
return {
 findById:function(name, id, res){…},
 find: function(name, query, res){…},
 put: function(name, res, req){…},
 delete: function(name, id, res){…}
}
```

DEMO

Code And Demo

https://react-blogg-server.herokuapp.com/gql

https://react-blogg-client.herokuapp.com/

https://github.com/maitriyogin/react-blogg-server

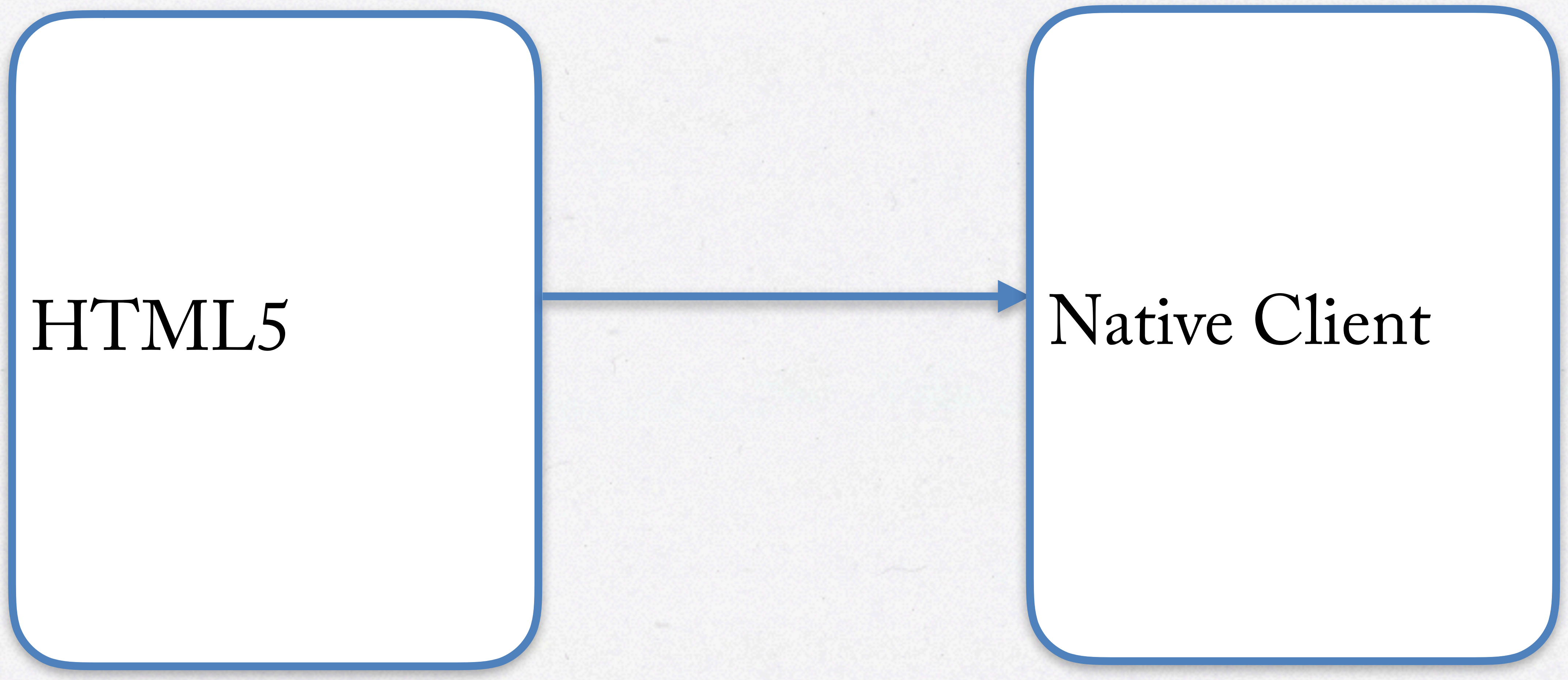
https://github.com/maitriyogin/react-blogg-client

Brief history

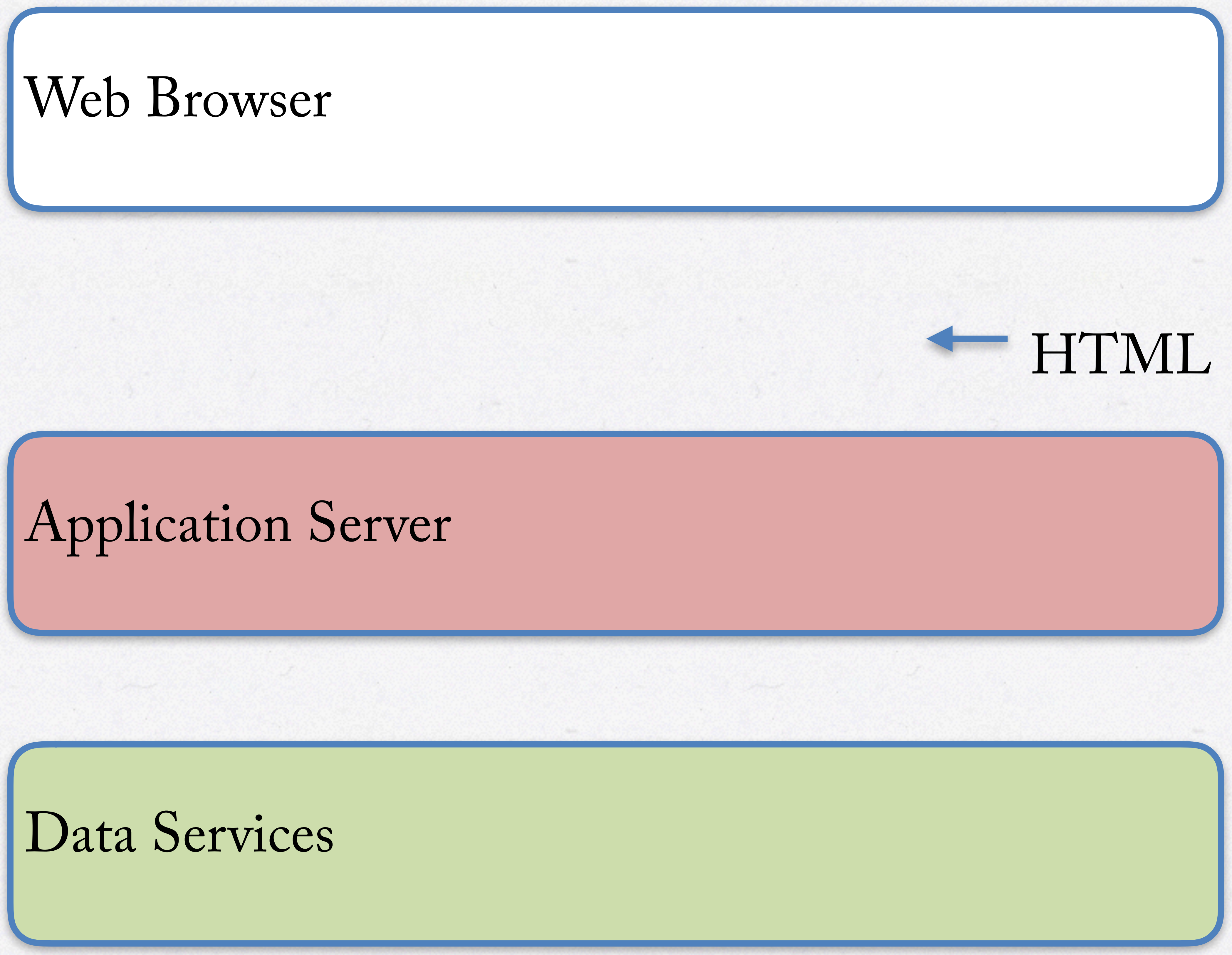
It started with a news feed

MOVE TO MOBILE

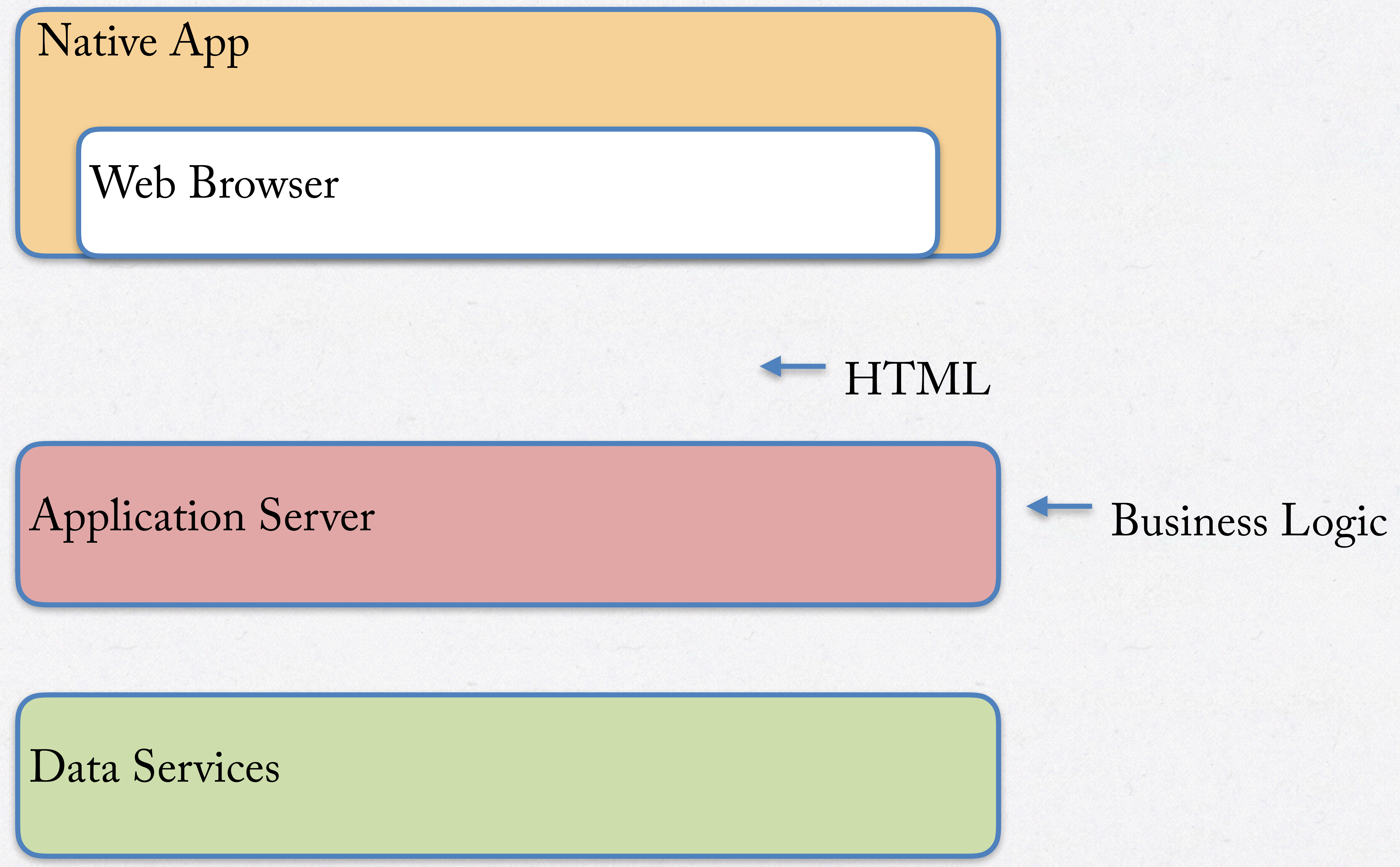
Facebooks move to Mobile 2011



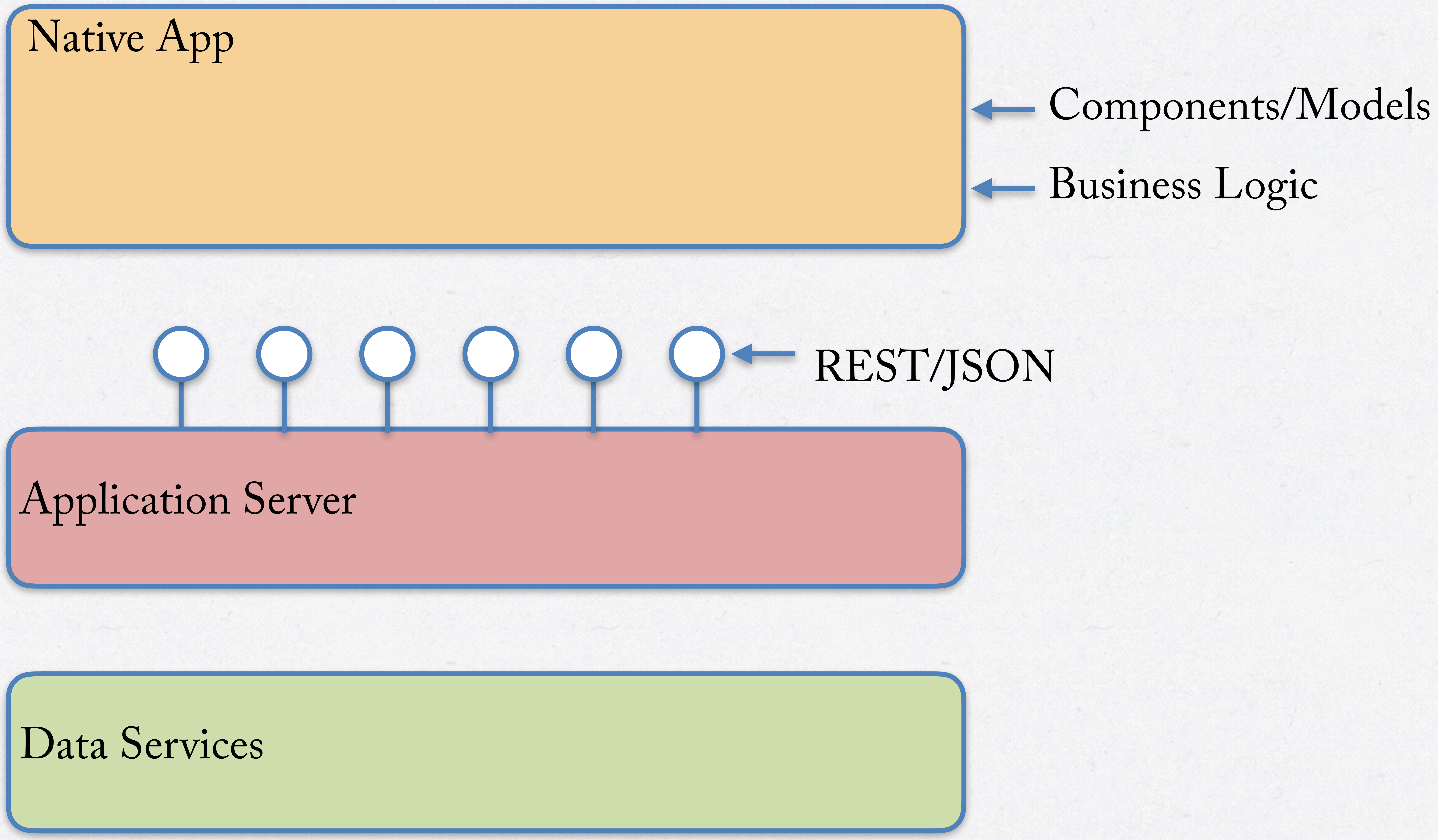
ORIGINAL STACK



NATIVE SHELL AROUND A WEB VIEW



NATIVE CLIENT WITH REST



MOVED TO REST

RESTful API

PROBLEMS THEY ENCOUNTERED WITH REST

Coupling : Tight

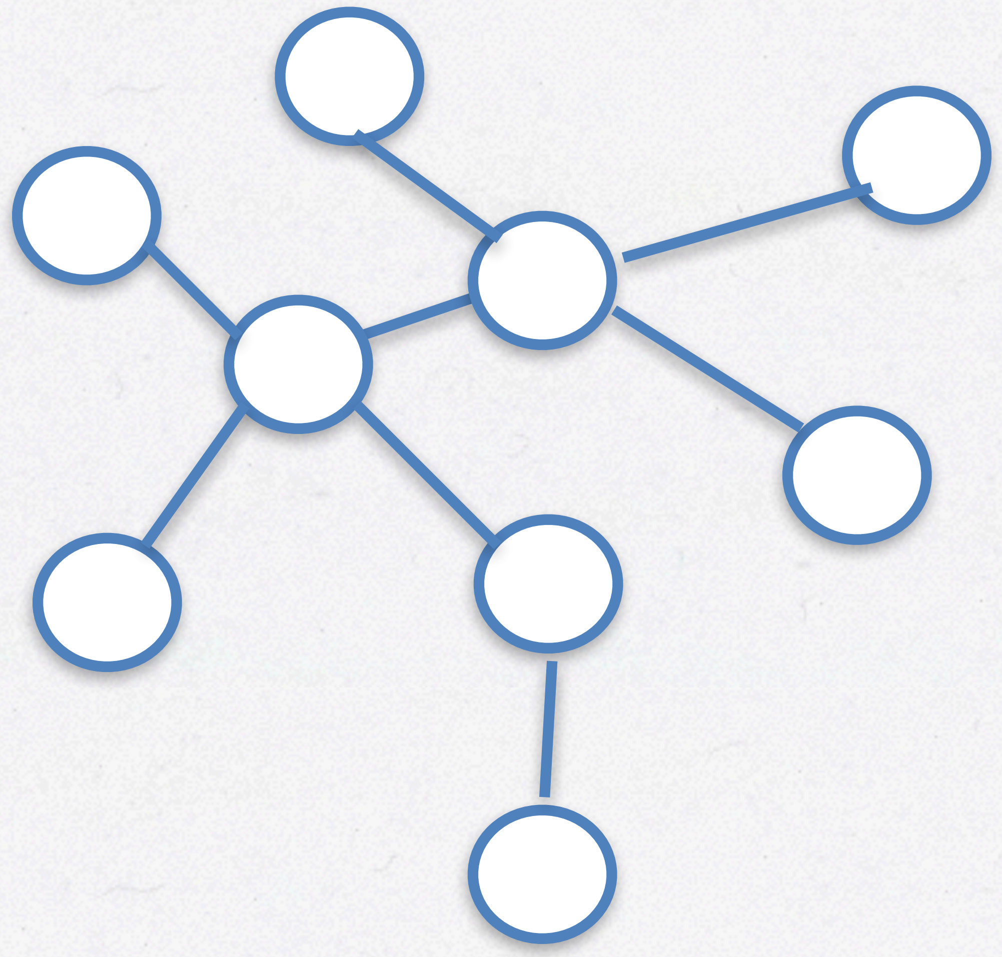
Cohesion: Low

MAIN ISSUES WITH REST, DRIVERS TO GRAPHQL

- **Multiple round trips for complicated object graphs**
- **Client Transformations**
- Too many **ad hoc endpoints**
- **Documentation** or specifications invariably become **outdated**.
- **REST** is intended for **long-lived network-based** applications that span **multiple organisations**…
  - **Not** really suited for an **API that serves a client app** built by the same organisation.

GRAPH-QL

Client first!



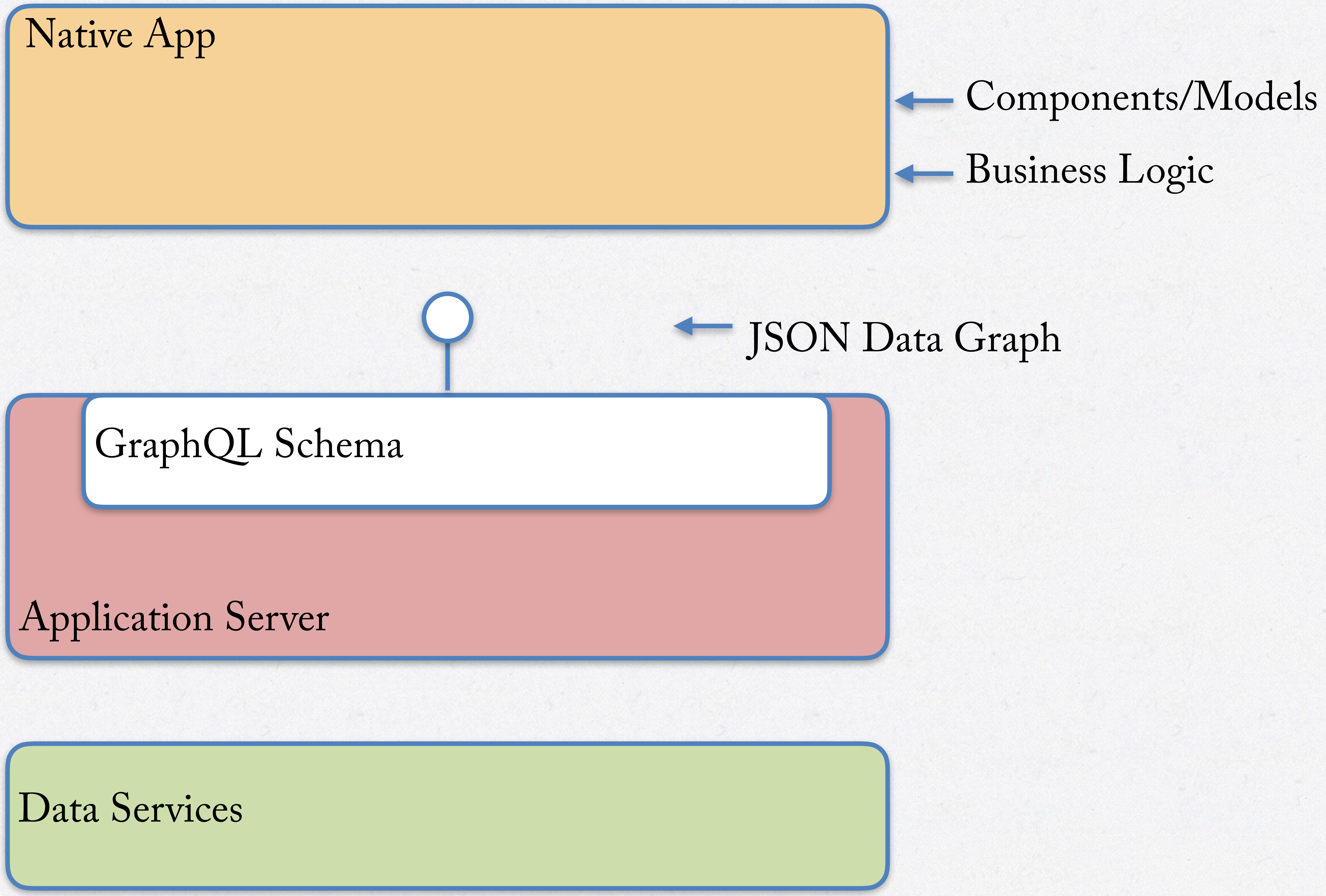
Product Centric

Graph of Data

Client decides the shape

Only get what you need

GRAPHQL ARCH



SUMMARY

- Composition : The component decides!
- The api/queries are decided by the client needs
- One endpoint
- Version agnosticism
- No need for multiple client transformation
- GraphIQL - self documenting api - very cool!
- Relay… next time

REST API

https://react-blogg-server.herokuapp.com/api/users

https://react-blogg-server.herokuapp.com/api/posts

https://react-blogg-server.herokuapp.com/api/comments