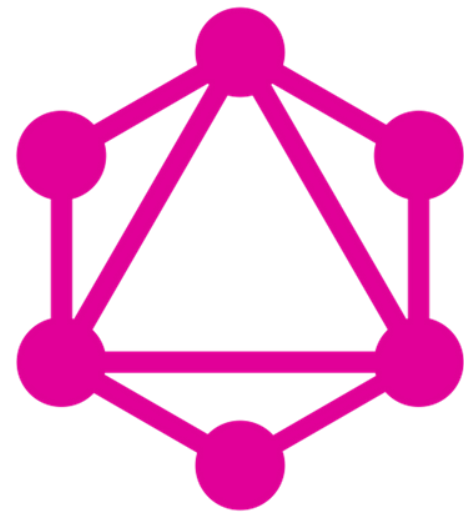


Otimizando buscadores de endereços com GraphQL

Patrick Silva Ferraz
<patrick.ferraz@outlook.com>

Rita Barretto
<rb.dccufba@gmail.com>

18 de setembro de 2019



GraphQL

—

Conceitos

HTTP

(Hypertext Transfer Protocol)

	Path to the source on Web Server	Protocol Version Browser supports
The HTTP Method	Post /RegisterDao.jsp HTTP/1.1	
The Request Headers	Host: www.javatpoint.com	
	User-Agent: Mozilla/5.0	
	Accept: text/xml,text/html,text/plain,image/jpeg	
	Accept-Language: en-us,en	
	Accept-Encoding: gzip,deflate	
	Accept-Charset: ISO-8859-1,utf-8	
	Keep-Alive:300	
	Connection:keep-alive	
User=ravi&pass=java		Message body

- O HTTP é o principal protocolo de comunicação para sistemas Web, existente há mais de 20 anos, e em todo esse tempo sofreu algumas atualizações.
- Nos anos 2000, um dos principais autores do protocolo HTTP, Roy Fielding, sugeriu o uso de novos métodos HTTP.
- Pretendiam resolver problemas relacionados à semântica quando requisições HTTP eram feitas.

API

(Application Programming Interface)

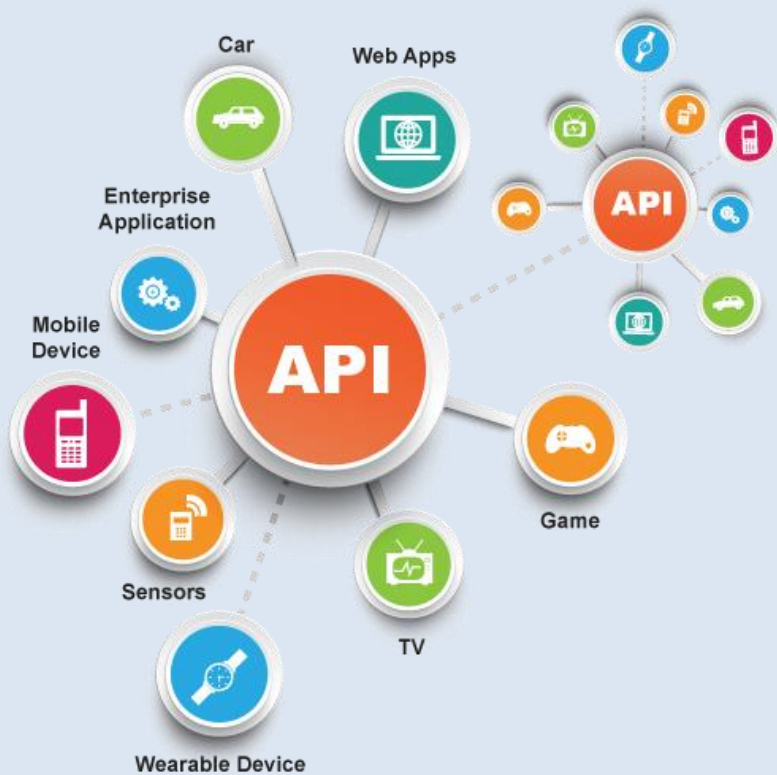


O que é API?

- **Conjunto de rotinas e padrões estabelecidos e documentados por uma aplicação A.** Para que outras aplicações consigam utilizar as funcionalidades desta aplicação A, **sem precisar conhecer detalhes da implementação do software.**

API

(Application Programming Interface)



O que é API?

- APIs permitem uma **interoperabilidade entre aplicações**.
 - Comunicação entre aplicações e entre os usuários.

API

(Application Programming Interface)



Representações

- Três exemplos:

Representação XML

```
1 <endereco>
2   <rua>
3     Rua Recife
4   </rua>
5   <cidade>
6     Paulo Afonso
7   </cidade>
8 </endereco>
```

Formato XML - Mais palavrosa, exigindo um esforço extra por parte de quem está escrevendo.

Representação JSON

```
1 { endereco:
2   {
3     rua: Rua Recife,
4     cidade: Paulo Afonso
5   }
6 }
```

Formato JSON - Mais leve de se escrever.

Representação YAML

```
1 endereco:
2 rua: rua Recife
3 cidade: Paulo Afonso
```

Formato YAML - Praticamente como escrevemos no dia a dia.

{ REST }

GET

/movies

Get list of movies

GET

/movies/:id

Find a movie by its ID

POST

/movies

Create a new movie

PUT

/movies

Update an existing movie

DELETE

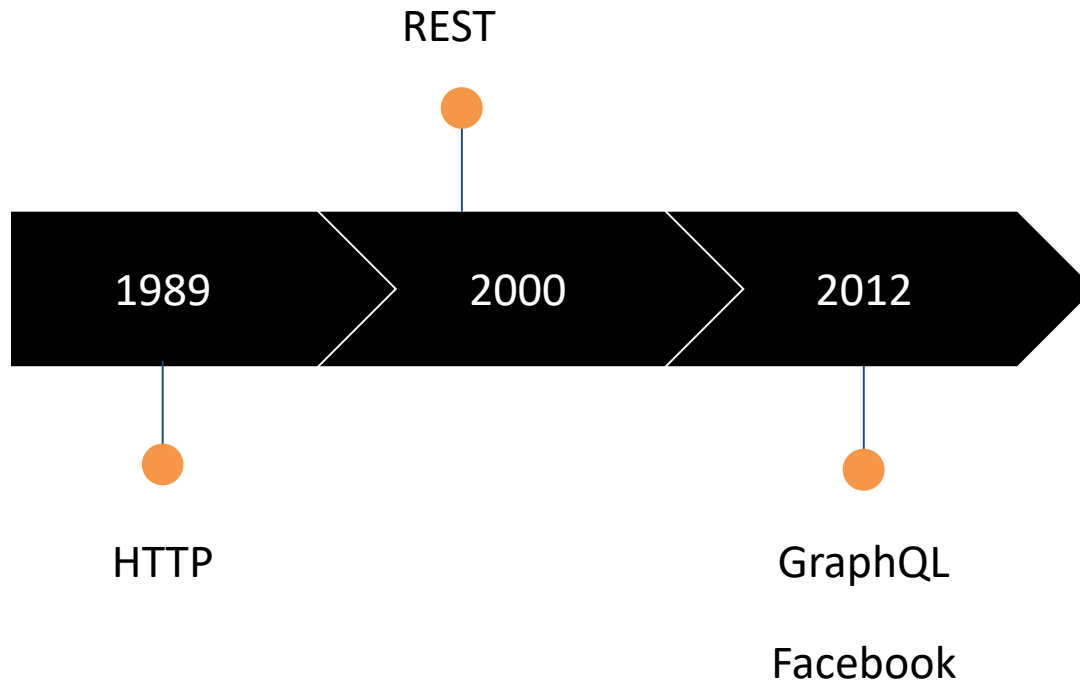
/movies

Delete an existing movie

O que é REST?

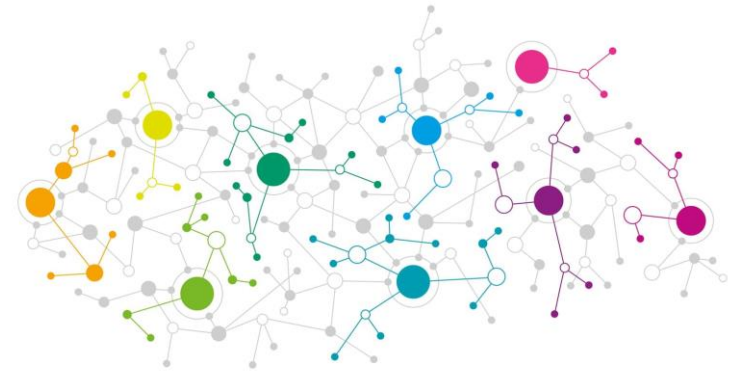
- **REST** significa *Representational State Transfer* (**Transferência de Estado Representacional**).
- **Abstração da arquitetura da Web.**
- Consiste em princípios/regras/constraints que permitem:
 - criação de um projeto com interfaces bem definidas;
 - aplicações se comuniquem.

Timeline





O que é GraphQL?



“O GraphQL é uma linguagem de consulta para APIs e um *runtime* para atender essas consultas com seus dados existentes. O GraphQL fornece uma descrição completa e compreensível dos dados em sua API, oferece aos clientes o poder de perguntar exatamente o que precisam e nada mais, facilita a evolução das APIs ao longo do tempo e permite ferramentas poderosas para desenvolvedores.”

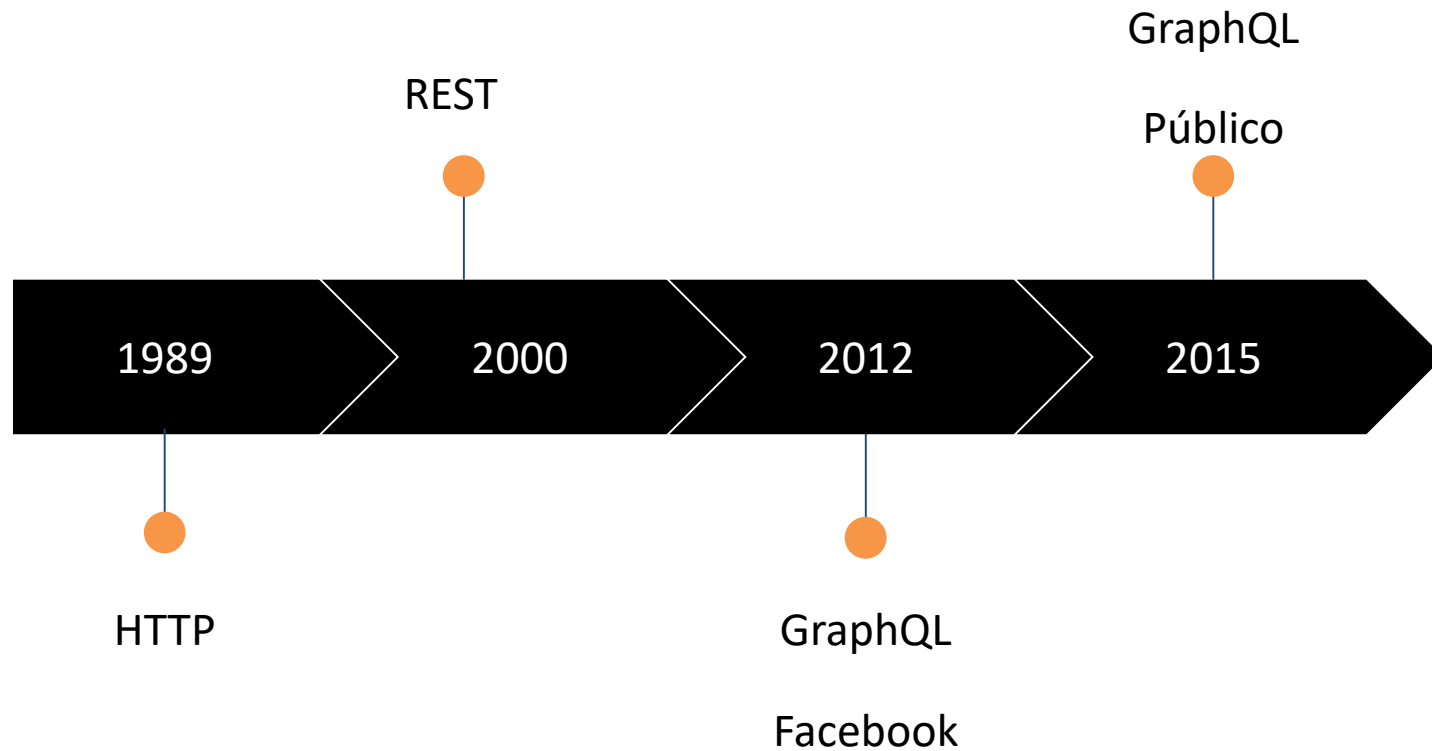
GraphQL.org



Para que serve?

- Para enviar uma consulta GraphQL para sua API e obter exatamente o que você precisa, nada mais e nada menos.
- Para dar rapidez e estabilidade aos aplicativos que usam o GraphQL porque controlam os dados que recebem, não o servidor.
- Permitir sites mais rápidos e fáceis de usar para os usuários e também para os desenvolvedores.

Timeline



REST vs GraphQL

Rest

Multiple round-trips

Multiple REST calls to get the data you need



Multiple Requests

Graph QL

Single round-trip

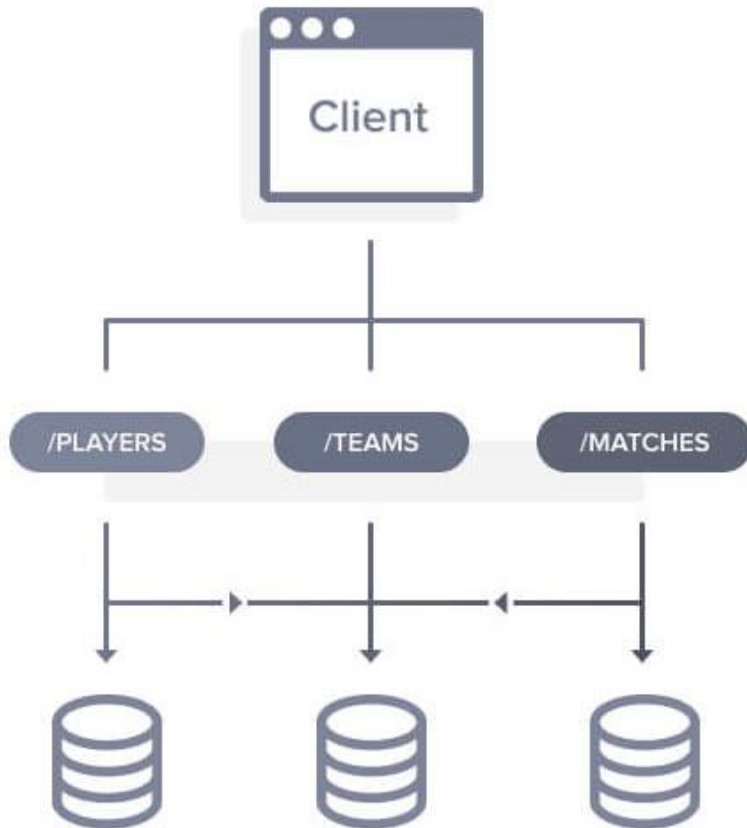
In a single network request :
query from multiple data source



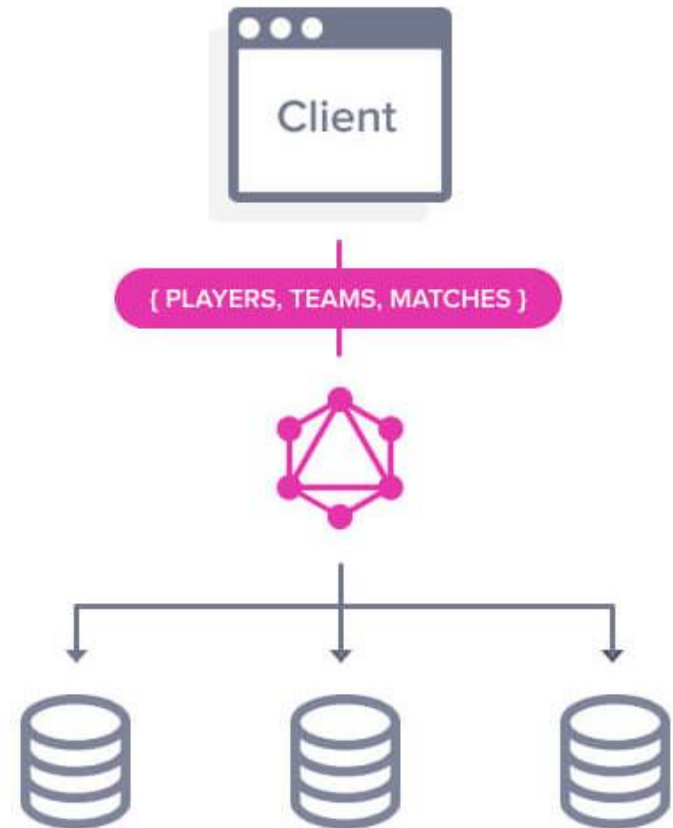
Single Request

Fonte: <https://www.multidots.com/graphql-efficient-alternative-rest/>

Rest API



GraphQL API



Fonte: <https://twitter.com/apollographql/status/1060226232124461056>

With REST



Feature development delayed, waiting on backend team

Hard to get consistency & feature parity across platforms

Challenging to move from monolithic backend
to microservices

Low confidence in the security of 100's of REST endpoints

Partners constantly need API customization or
use your API improperly

With GraphQL



Ship apps across platforms months faster

Consistent high-quality UX across all platforms

Ease transition to microservices

Centrally manage and secure your entire API

Flexible public API that enables new applications

Fonte: <https://everis.passle.net/post/102fjja/graphql-is-it-the-new-rest-part-4-of-4>

Quem utiliza?

FIRST²
LOOK
MEDIA



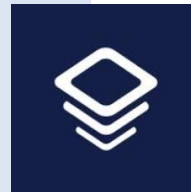
6



Casos de Uso



4



8



7



3



1

The New York Times

1 <https://open.nytimes.com/react-relay-and-graphql-under-the-hood-of-the-times-website-redesign-22fb62ea9764>

2 <https://code.firstlook.media/welcome>

3 <https://labs.mlssoccer.com/implementing-graphql-at-major-league-soccer-ff0a002b20ca>

4 <http://artsy.github.io/blog/2016/11/02/improving-page-speed-with-graphql/>


5 <https://engineeringblog.yelp.com/2017/05/introducing-yelps-local-graph.html>

6 <https://githubengineering.com/the-github-graphql-api/>

7 <https://help.shopify.com/en/api/custom-storefronts/storefront-api/graphql>

8 <https://fabric.io/blog/building-fabric-mission-control-with-graphql-and-relay>

Onde posso usar?



Linguagens de Programação

- C# / .NET
- Clojure
- Elixir
- Erlang
- Go
- Groovy
- Java
- JavaScript
- PHP
- Python
- Scala
- Ruby



Frameworks

- JavaScript
 - express
 - apollo
- Python
 - django
 - flask
- Databases
 - mongo
 - sqlalchemy
- Clients...



Frameworks

- JavaScript
 - express
 - apollo
- Java
 - spring
- **Python**
 - django
 - **flask**
- Clients...

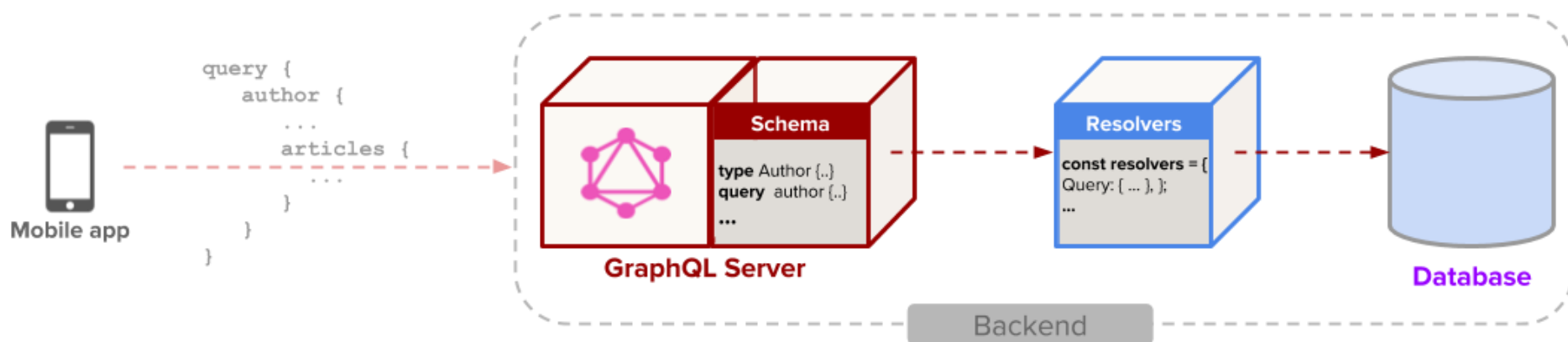
E os bancos de dados?



—

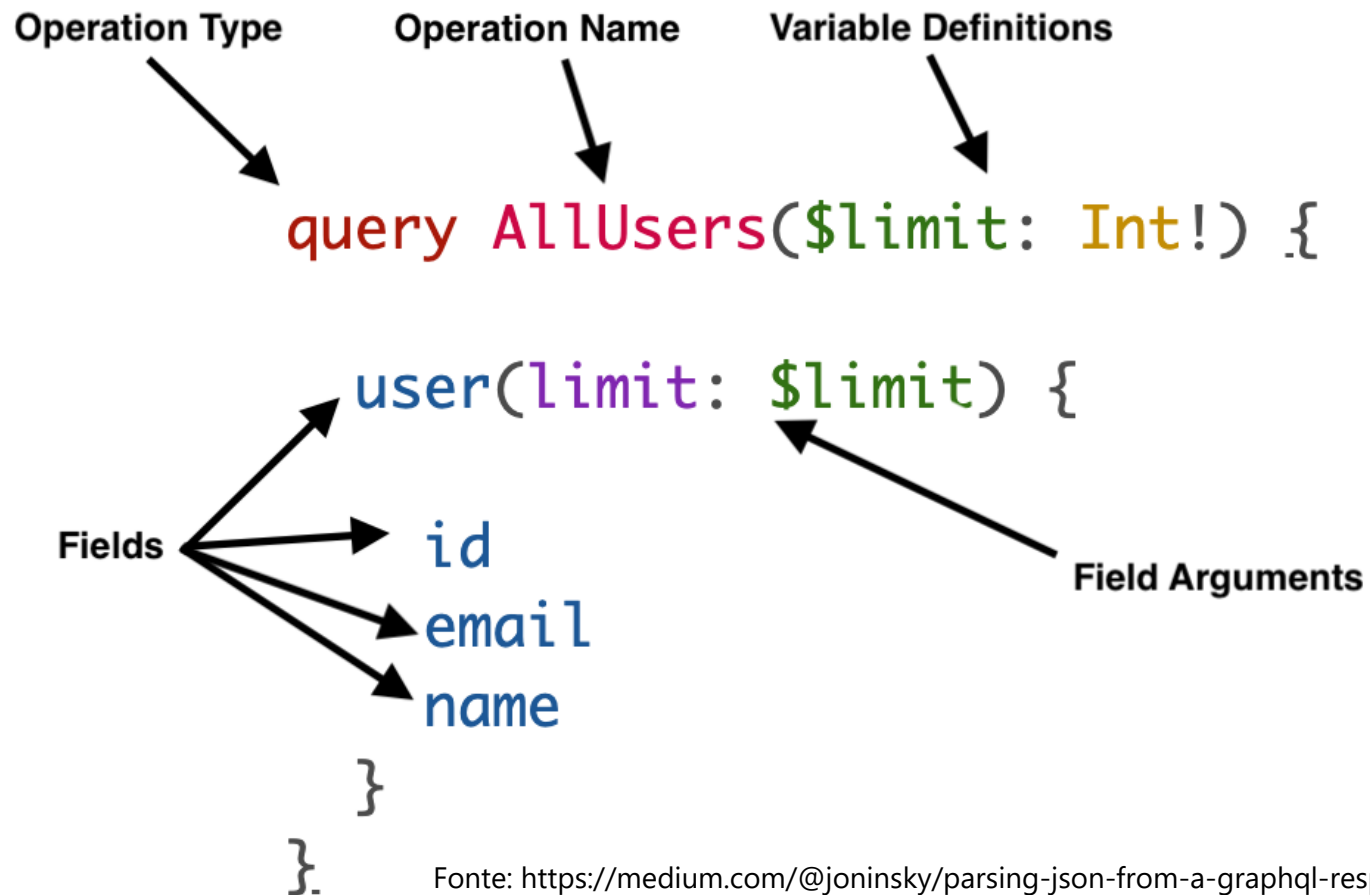
Arquitetura

Arquitetura



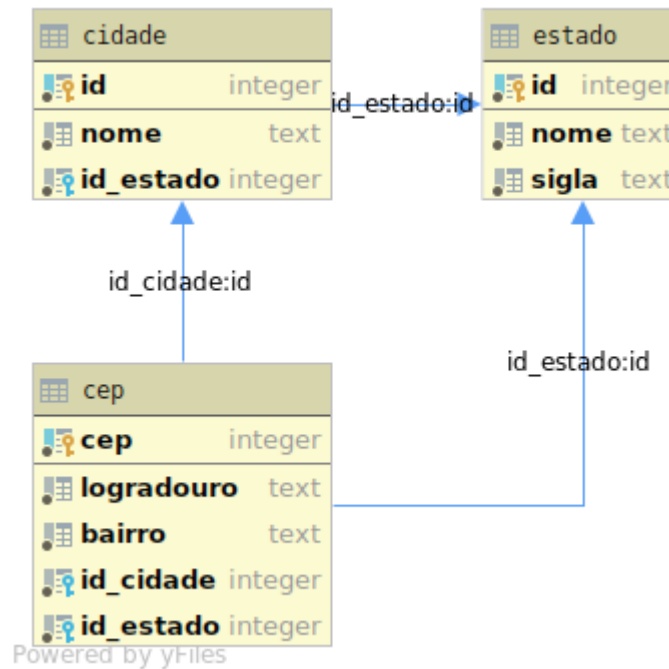
Fonte: <https://www.programmableweb.com/news/top-5-things-to-remember-when-adding-graphql-backend/analysis/2018/10/01>

Estrutura JSON



Fonte: <https://medium.com/@joninsky/parsing-json-from-a-graphql-response-854e8a29afef>

DER do Database



GraphQL Server

Fluxo de operações do GraphQL

Descrever seus dados

```
type State {  
  id: String  
  name: String  
  acronym: String  
  cities: [City]  
}  
type City {  
  id: String  
  id_state: String  
  name: String  
}
```

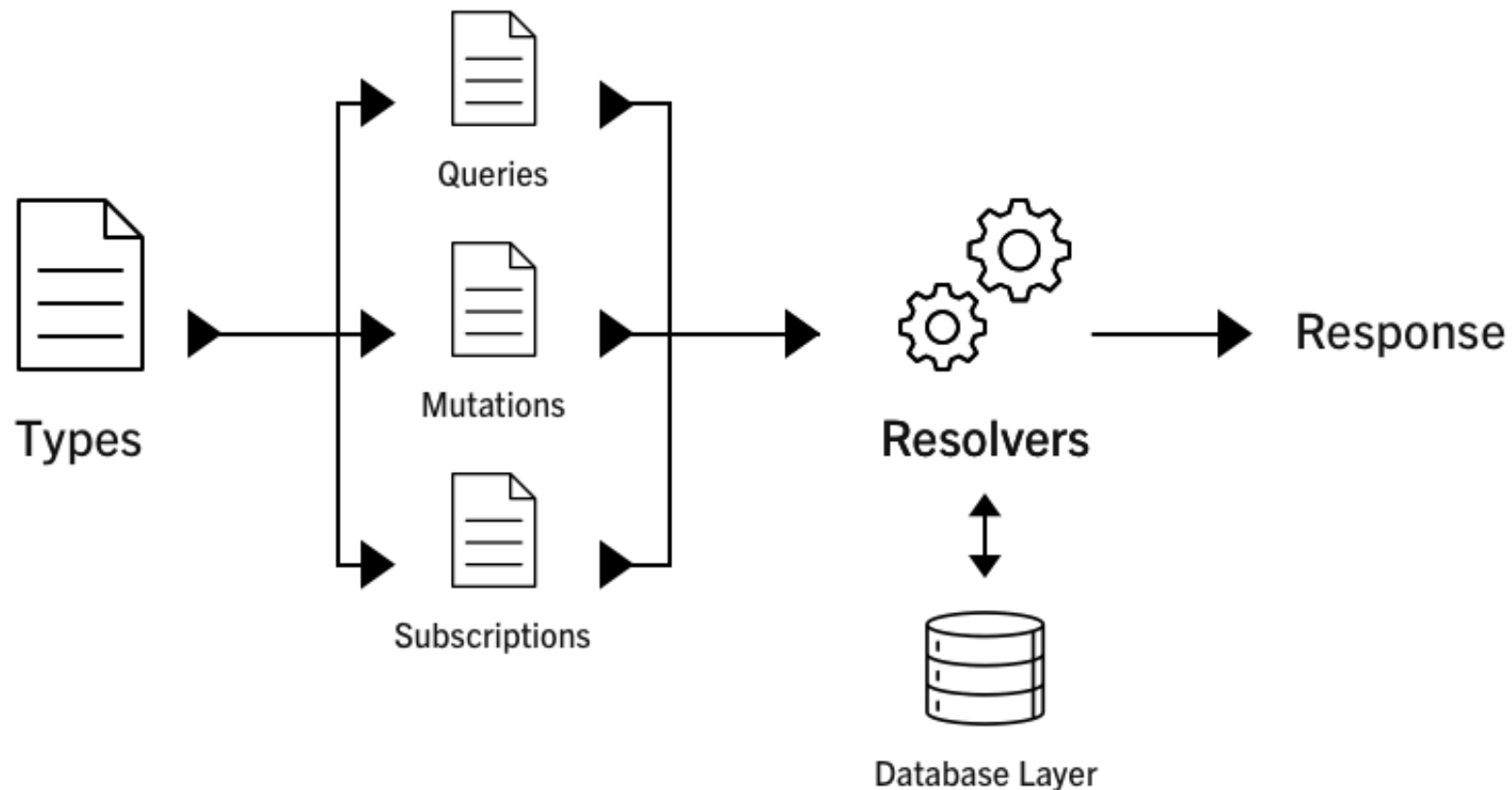
Pedir o que você quer

```
{  
  state(name: "bahia") {  
    cities {  
      name  
    }  
  }  
}
```

Obter resultados previsíveis

```
{  
  "state": {  
    "cities": [  
      {"name": "Salvador"},  
      {"name": "Camaçari"},  
      ...  
    ]  
  }  
}
```

Estrutura do GraphQL



Fonte: <https://hackernoon.com/creating-a-structured-hot-reloadable-graphql-api-with-express-js-de62c859643>



Type System

- Scalar Types

- String
- Int
- Float
- Boolean
- ID
- Date
- DateTime
- Time
- JSONString



Type System

- Scalar Types

Argumentos:

- name
- description
- required
- deprecation_reason
- default_value

CODE



Type System

- Scalar Types
- Enums

Tipos enumerados

- Validar algum argumento
- Comunica o valor de um campo



Type System

- Scalar Types
- Enums

```
enum Episode {  
    NEWHOPE  
    EMPIRE  
    JEDI  
}
```

(NEWHOPE = 0, EMPIRE = 1, JEDI = 2)

CODE



Type System

- Scalar Types
 - Enums
 - Lists and Non-Null
- Modificadores de tipo
 - Afeta a validação dos valores

CODE



Schema Language

- Fields
 - Scalar types
 - Arguments
- Field resolvers
 - Arguments
 - Actions



Schema Language

- ObjectTypes

- Componente mais básico
- Define um tipo de objeto
- Como obtê-lo
- Que campos possui

CODE



Schema Language

- ObjectTypes
- AbstractTypes

- Fields podem ser compartilhados com:
 - ObjectType
 - Interface
 - InputObjectType
 - AbstractType

CODE



Schema Language

- ObjectTypes
 - AbstractTypes
 - Interfaces
- É um abstract type, com algumas diferenças
 - Conjunto de fields devem ser compartilhados

CODE



Schema Language

- ObjectTypes
 - AbstractTypes
 - Interfaces
 - Unions
- Semelhante as interfaces
 - Não especificam fields
 - Possuem links para as fields possíveis

CODE



Schema Language

- ObjectTypes
- AbstractTypes
- Interfaces
- Unions
- InputObjectType

- Um pouco dos dois mundos (types | schema)
- **Object** vs **Scalar**

CODE



Schema Language

Queries

OHH NO
SPOILER
ObjectType



Schema Language

Mutations

- **ObjectType** especial que também define uma entrada

CODE

—

Resumindo

GraphQL Schema

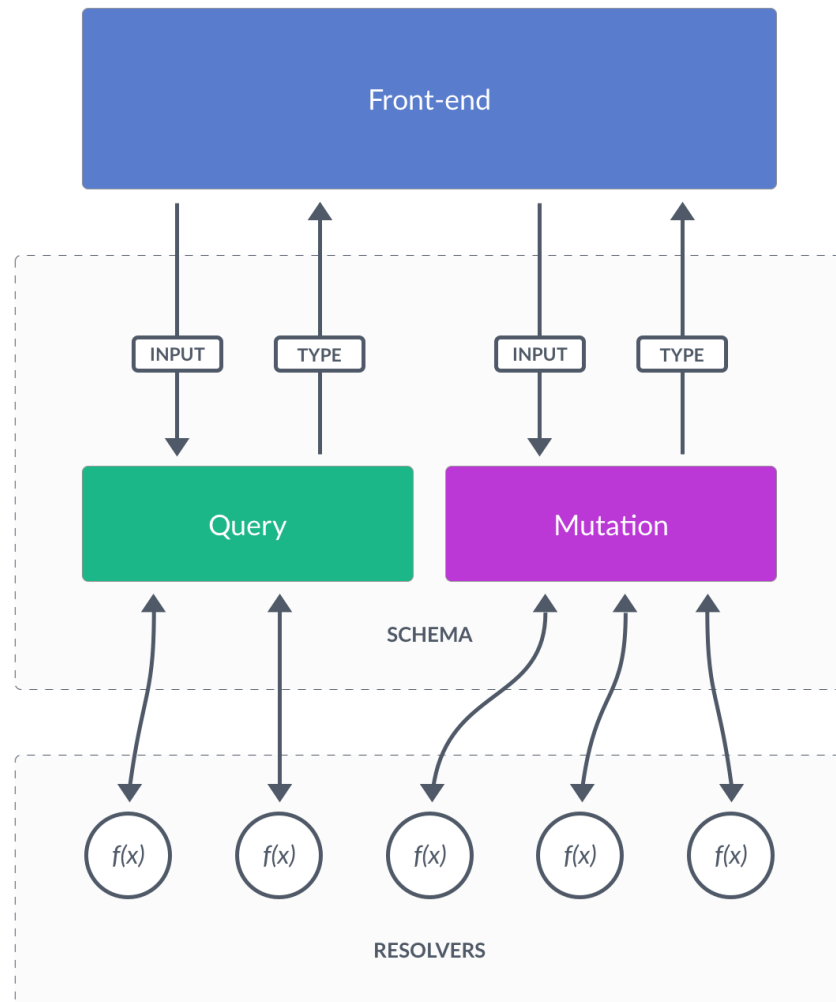
```
type User {  
  id: ID!  
  name: String!  
}  
  
type Query {  
  users: [User]!  
}  
  
type Mutation {  
  createUser(name: String!): User!  
}
```

Available GraphQL Operations

```
query {  
  users {  
    id  
    name  
  }  
}  
  
mutation {  
  createUser(name: "Sarah") {  
    id  
  }  
}
```

● Root Type ● Root Field ● Operation Type

Selection Set

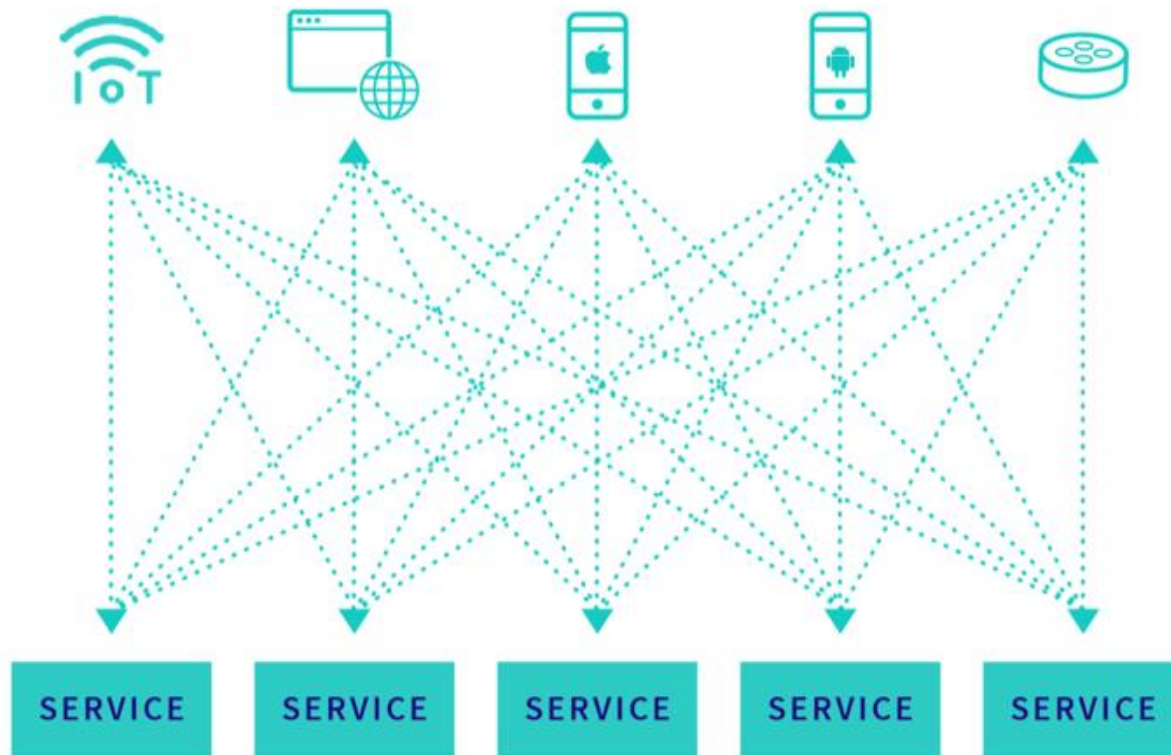


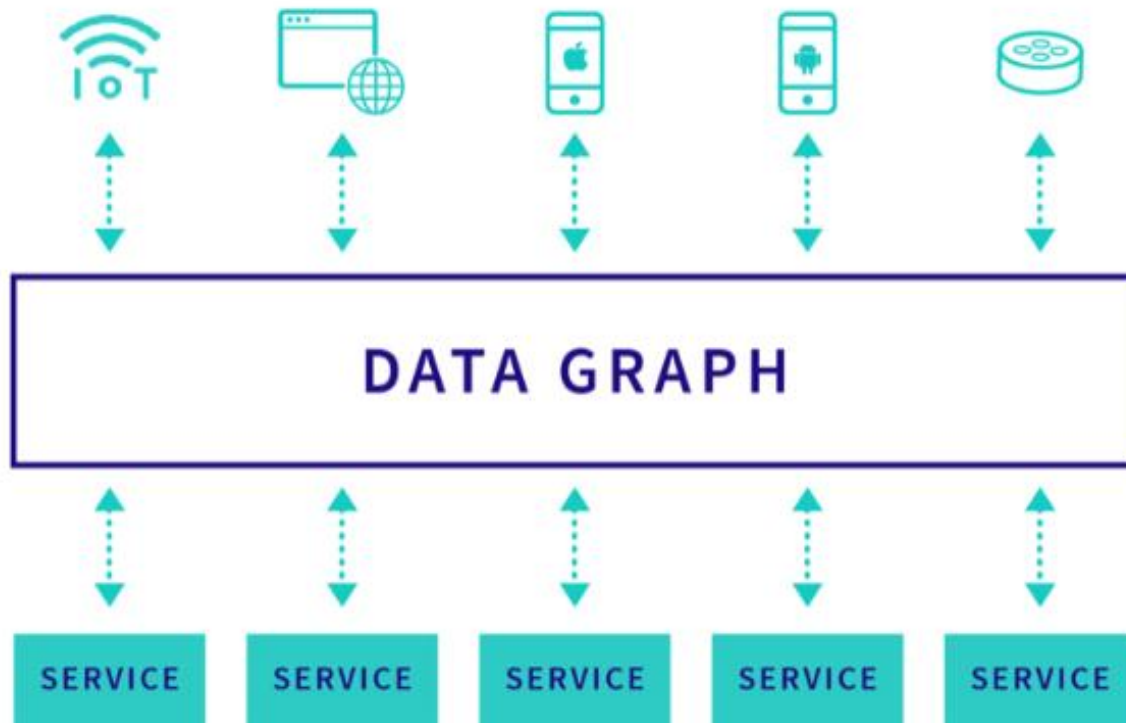
EXECUTE API

PYTHON + FLASK + GRAPHQL



Next Steps





DÚVIDAS

